SQL-Befehlsliste

Vereinbarung über die Schreibweise

	Beschreibung	Beispiele
SchlüsselWort	Befehlsworte in SQL-Anweisungen werden fett und in Großbuchstaben geschrieben	**SELECT ... FROM ... ;**
[optionale Elemente]	mögliche, aber nicht zwingend erforderliche Teile von Anweisungen stehen [in eckigen Klammern]	[**WHERE** *Bedingung*]

SQL-Datentypen

numerische Datentypen	**Beschreibung**	**Speicherplatz**
DEC[(M,D)], DECIMAL[(M,D)] NUMERIC[(M,D)]	exakte Festkommazahl - mit M Stellen (ohne Dezimaltrennzeichen), - davon D Dezimalstellen	
DOUBLE[(M,D)], REAL[(M,D)]	Fließkommazahl mit doppelter Genauigkeit - mit M Stellen, - davon D Dezimalstellen	**8 Byte**
FLOAT[(M,D)]	Fließkommazahl mit einfacher Genauigkeit - mit M Stellen, - davon D Dezimalstellen	**4 Byte**
INT, INTEGER	ganze Zahlen - ohne Vorzeichen: von 0 bis 0 4.294.967.295 - mit Vorzeichen : von –2.147.483.648 bis 2.147.483.647	**4 Byte**
String-Typen	**Beschreibung**	**Speicherplatz**
CHAR(M)	Zeichenkette mit fester Länge (M = 1 bis 255)	
VARCHAR(M)	Zeichenkette mit variabler Länge (M = 1 bis 65.532)	
Zeit-Datumstypen	**Beschreibung**	**Speicherplatz**
DATE	liefert Werte, die nur das Datum enthalten: **'YYYY-MM-DD'**	
DATETIME	liefert Werte, die sowohl ein Datum als auch eine Uhrzeit enthalten: **'YYYY-MM-DD HH:MM:SS'**	
boolsche Typen	**Beschreibung**	**Speicherplatz**
TINYINT	wird als boolscher Datentyp behandelt. 0 bedeutet falsch (false), 1 oder >1 bedeutet wahr (true)	1 Byte
weitere Attribute:	**Beschreibung**	**Speicherplatz**
AUTO_INCREMENT	Der Wert dieses Attributs wird automatisch beim Anlegen eines neuen Datensatzes aus dem Wert des Datenfeldes des vorherigen Datensatzes + 1 berechnet. Nur bei INT-Typen	
DEFAULT (Wert)	Definiert einen Standardwert für dieses Feld	
NOT NULL	Die Eingabe eines Wertes für dieses Datenfeld wird erzwungen.	
NULL	Das Datenfeld hat standardmäßig keinen Wert	

SQL-Statements

SQL	Beschreibung/Syntax/Beispiel
SELECT... FROM	Auswahl von Datensätzen und Feldern aus einer Datenbank Syntax: SELECT [ALL/DISTINCT] {Spalten/*} FROM tabelle [,tabelle,...] [WHERE Bedingung] [GROUP BY Spalten [HAVING Bedingung]] [ORDER BY Spalten [ASC/DESC]]; Erklärung: DISTINCT vermeidet die Auswahl doppelter/identischer Datensätze z.B.: SELECT teile.teilenr, teile.bezeichnung FROM teile;
BETWEEN ... AND	Bestimmt, ob der Wert eines Ausdrucks innerhalb eines bestimmten Bereichs von Werten liegt. z.B.: SELECT * FROM teile WHERE ekpreis BETWEEN 20 AND 50;

SQL-Befehlsliste

CREATE DATABASE ...	Erstellt eine Datenbank Syntax: CREATE DATABASE [IF NOT EXISTS] Datenbankname; z.B.: CREATE DATABASE IF NOT EXISTS fahrradvermietung;
CREATE TABLE ...	Erstellt eine Tabelle Syntax: CREATE TABLE Tabellenname (Datenfeld1 Datentyp [Attribut1 Attribut2], Datenfeld2 Datentyp [Attribut1 Attribut2], PRIMARY KEY (Datenfeld), FOREIGN KEY (Datenfeld) REFFERENCES Datenbankname.Tabellenname (Datenfeld) [ON DELETE {RESTRICT | CASCADE | SET NULL | NO ACTION}] [ON UPDATE {RESTRICT | CASCADE | SET NULL | NO ACTION}]) ENGINE = INNODB;
DELETE FROM	löscht Datensätze in einer oder mehreren Tabellen einer Datenbank Syntax: DELETE FROM Tabelle WHERE Bedingung; z.B.: DELETE FROM kunden WHERE kunden.kdnr = "1241";
DROP DATABASE	löscht die angegebene Datenbank Syntax: DROP DATABASE Datenbankname;
GROUP BY	Fasst Datensätze, die in der angegebenen Feldliste dieselben Werte enthalten, zu einem einzelnen Datensatz zusammen. Für jeden Datensatz wird ein Ergebniswert berechnet, wenn Sie eine SQL-Aggregatfunktion wie Sum oder Count in der SELECT-Anweisung angeben. z.B.: SELECT plz, SUM(umsatz) AS Umsätze FROM lieferer GROUP BY plz;
HAVING	Gibt an, welche der gruppierten Datensätze in einer SELECT-Anweisung mit einem GROUP BY-Abschnitt angezeigt werden sollen. Nachdem GROUP BY Datensätze kombiniert, zeigt HAVING alle vom GROUP BY-Abschnitt gruppierten Datensätze an, die die im HAVING-Abschnitt angegebenen Bedingungen erfüllen. z.B.: ... FROM lieferer GROUP BY plz HAVING SUM(umsatz) > 30000;
IN	Gibt die Datensätze zurück, die in dem in der Abfrage genannten Feld einen aus einer Auflistung von Werten beinhalten. z.B.: ... WHERE teile.teileart in ('E', 'T', 'R');
INNER JOIN ... ON	Kombiniert Datensätze aus zwei Tabellen, sobald ein gemeinsames Feld dieselben Werte hat. Syntax: FROM Tabelle1 INNER JOIN Tabelle2 ON Tabelle1.Feld1 = Tabelle2.Feld2 z.B.: FROM bestpos INNER JOIN teile ON bestpos.teilenr = teile.teileNr;
INSERT INTO	Fügt einer Tabelle einen (oder mehrere) Datensätze hinzu Syntax: INSERT INTO Tabelle [(Spalte[,Spalte,..)] VALUES (Wert[,Wert,...]; z.B.: INSERT INTO kunden (name, vorname) VALUES ('Maier', 'Rudolf');
IS [NOT] NULL	Prüft, ob ein Attribut [nicht] leer ist. z.B.: SELECT * FROM hersteller where portal IS NULL; (liefert alle Datensätze,.bei denen das Attribut portal leer ist)
LIKE	Dient zum Vergleichen zweier Zeichenfolgen. z.B.: SELECT name FROM lieferer WHERE plz LIKE "7%"; (alle Lieferer-Datensätze des Postleitbereichs 7 werden ausgewählt)
ORDER BY	Sortiert die Daten eines Recordset-Objekts nach einem oder mehreren angegebenen Feldern in aufsteigender oder absteigender Reihenfolge. z.B.: SELECT liefnr, name FROM lieferer ORDER BY name ASC; (aufsteigend) oder SELECT liefnr, name FROM lieferer ORDER BY name DESC; (absteigend)
PRIMARY KEY (Feld)	Weist einem Datenfeld die Eigenschaft "Primärschlüssel" zu. z.B.: PRIMARY KEY (kdnr)

SQL-Befehlsliste

SELECT ... INTO	erstellt eine neue Tabelle Syntax: SELECT {Spalte[,Spalte...]/*} INTO Tabelle FROM Tabelle [WHERE Bedingung]; z.B.: SELECT * INTO handelswaren FROM teile WHERE teile.teileart = 'H';
SHOW DATABASES	listet alle vorhandenen Datenbanken auf Syntax: SHOW DATABASES;
UPDATE ... SET	ändert Werte in Feldern einer Tabelle Syntax: UPDATE Tabelle SET Spalte = Ausdruck [,Spalte = Ausdruck ...] [WHERE Bedingung]; z.B.: UPDATE Teile SET EKPreis = EKPreis * 1.1;
USE	Wählt eine Datenbank aus, die bearbeitet werden soll. Syntax: USE Datenbankname; z.B.: USE fahrradvermietung;
WHERE	Der Teil einer SQL-Anweisung, der angibt, welche Datensätze abgerufen werden sollen. Der WHERE-Abschnitt beschränkt den Umfang der Abfrage (Selektion). z.B.: SELECT * INTO handelswaren FROM teile WHERE teile.teileart = 'H'; SELECT * FROM artikel WHERE wg IS NULL;

SQL-Funktionen

AVG()	Berechnet den arithmetischen Mittelwert einer Menge von Werten in einem bestimmten Feld einer Abfrage. Syntax: AVG(Ausdr) z.B.: SELECT AVG(umsatz) AS Durchschnitt FROM lieferer;
COUNT()	Berechnet die Anzahl der von einer Abfrage zurückgegebenen Datensätze. Syntax: COUNT(Ausdruck) z.B.: SELECT COUNT(*) AS Anzahl FROM teile WHERE ekpreis > 100;
DATE()	Extrahiert den Datumsteil aus dem DATE- oder DATETIME-Ausdruck Syntax: DATE(Ausdruck) z.B.: SELECT * FROM auftrag WHERE aufdat = date(now());
DATEDIFF()	Berechnet die Anzahl Tage zwischen dem Startdatum und dem Enddatum Syntax: DATEDIFF(Ausdruck1, Ausdruck2) z.B.: SELECT DATEDIFF(bis, von) AS Dauer FROM vermietungen;
MAX()	Gibt den größten Wert aus einer Reihe von Werten zurück, die in einem bestimmten Feld einer Abfrage enthalten sind. Syntax: **Max**(Ausdr) z.B.: SELECT MAX(umsatz) AS 'höchster Umsatz' FROM lieferer;
MIN()	Gibt den kleinsten Wert aus einer Reihe von Werten zurück, die in einem bestimmten Feld einer Abfrage enthalten sind. Syntax: **Min**(Ausdr) z.B.: SELECT MIN(umsatz) AS 'kleinster Umsatz' FROM lieferer;
NOW()	Liefert das aktuelle Tagesdatum. Syntax: NOW() z.B.: SELECT * FROM auftrag WHERE aufdat = now();
SUM()	Berechnet die Summe einer Menge von Werten in einem bestimmten Feld einer Abfrage. z.B.: SELECT SUM(umsatz) AS Liefererumsätze FROM lieferer;
TIMEDIFF()	Berechnet den Zeitraum zwischen der Startzeit datum1 und der Endzeit datum2. Syntax: TIMEDIFF(datum1, datum2) z.B.: SELECT TIMEDIFF(bis, von) AS Dauer FROM vermietungen;
YEAR()	Liefert das Jahr eines Datums. z.B. SELECT year(now()) - year(gebdat) FROM teilnehmer;

SQL-Befehlsliste

SQL-Tansaktionskontrolle

COMMIT	Abschluss einer Transaktion Syntax: COMMIT;
LOCK	
ROLLBACK	Setzt die Datenbank auf den Zustand vor Beginn der Transaktion oder auf einen Transaktions-Speicherpunkt zurück Syntax: ROLLBACK [Bezeichner]; z. B.:
SAVEPOINT	Legt einen Transaktions-Speicherpunkt fest Syntax: SAVEPOINT Bezeichner; z. B.:
SET AUTOCOMMIT	
START TRANSACTION	Beginn einer Transaktion Syntax: START TRANSACTION;
UNLOCK	

SQL-Benutzer- und Rechteverwaltung

CREATE USER	Legt einen Benutzer an. Syntax: `CREATE USER 'Name'@'Host;` z. B.: `CREATE USER 'admin'@'%';` (% = localhost)
SET PASSWORD	Legt das Passwort für einen Benutzer fest. Syntax: `SET PASSWORD FOR 'Name'@'Host' = PASSWORD('Kennwort');` z.B.: `SET PASSWORD FOR 'admin'@'%'= PASSWORD('admin');`
GRANT	Gewährt Rechte auf alle Tabellen Syntax: `GRANT ALL PRIVILEGES ON *.* TO <Name>@<Host>;` z. B.: `GRANT ALL PRIVILEGES ON *.* TO 'admin'@'%'`
REVOKE	Entzieht Rechte auf Tabellen. Syntax: `REVOKE ALL PRIVILEGES ON *.* FROM <Name>@<Host>;` z. B.: `REVOKE ALL PRIVILEGES ON *.* FROM 'Hugo'@'%'`