

H9 SQL -- DML.

Insert Update Delete





Insert.



SQL – DML basisopdrachten

SQL basis opdrachten

SELECT raadplegen van gegevens

zie vorig hoofdstuk

INSERT toevoegen van gegevens

UPDATEwijzigen van gegevens

DELETE
 verwijderen van gegevens



Toevoegen van data -- INSERT

- Toevoegen van rijen in een tabel gebeurt via het INSERT statement:
 - één enkele rij toevoegen via specificatie van waarden
 - geselecteerde rij(en) uit een andere tabel toevoegen
 (2TI)

```
INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE] [INTO] tbl_name [(col_name [, col_name] ...)] {VALUES | VALUE} (value_list) [, (value_list)] ... [ON DUPLICATE KEY UPDATE assignment_list]
```



INSERT van één rij

 <u>Voorbeeld</u>: Voeg in de tabel Categories de categorie "Chocolade" toe met categoryID= 10

- methode 1: enkel de (niet NULL) waarden voor specifieke

kolommen worden opgegeven

INSERT INTO categories (categoryID, categoryName)
VALUES (10, 'Chocolade')

CategoryID CategoryName Description Soft drinks, coffees, teas, beers, and ... Beverages Condiments Sweet and savory sauces, relishes, sp... Confections Desserts, candies, and sweet breads Dairy Products Cheeses Grains/Cereals Breads, crackers, pasta, and cereal Meat/Poultry Prepared meats Produce Dried fruit and bean curd Seaweed and fish Seafood

Table: categories

CategoryID

Description

Picture

CategoryName

int(11) AI PK

varchar(15)

longtext

longblob

Columns:

- methode 2: alle kolomwaarden worden opgegeven

INSERT INTO categories

VALUES (10, 'Chocolade', NULL, NULL)

CategoryID	CategoryName	Description	
1	Beverages	Soft drinks, coffees, teas, beers, and \dots	
2	Condiments	Sweet and savory sauces, relishes, sp	
3	Confections	Desserts, candies, and sweet breads	
4	Dairy Products	Cheeses	
5	Grains/Cereals	Breads, crackers, pasta, and cereal	
6	Meat/Poultry	Prepared meats	
7	Produce	Dried fruit and bean curd	
8	Seafood	Seaweed and fish	
10	Chocolade	NULL	

INSERT van één rij

Het aantal opgegeven kolomnamen en waarden moeten gelijk zijn.

Het **type** van de waarde moet overeenstemmen met het datatype van de desbetreffende kolom.

Als **geen kolomnamen** worden opgegeven, worden de waarden toegekend volgens de **kolomvolgorde** zoals bepaald bij de definitie van de tabel (CREATE).

Ook NULL mag als waarde worden opgegeven.

Verplichte velden moet je opgeven, tenzij ze een default waarde

bevatten.

. . .

	Column Name	Data Type	Allow Nulls
₽Ÿ	ProductID	int	
	ProductName	nvarchar(40)	
	SupplierID	int	✓
	CategoryID	int	~
	QuantityPerUnit	nvarchar(20)	~
	UnitPrice	money	V
	UnitsInStock	smallint	✓
	UnitsOnOrder	smallint	✓
	ReorderLevel	smallint	✓
	Discontinued	bit	



INSERT van één rij

 Voorbeeld: Voeg gegevens toe aan de Customers tabel Table: customers

Columns:

CustomerID char(5) PK CompanyName varchar(40) ContactName varchar(30) ContactTitle varchar(30) Address varchar(60) City Region varchar(15) PostalCode varchar(10) Country varchar(15) Phone varchar(24) varchar(24) Fax

INSERT INTO customers

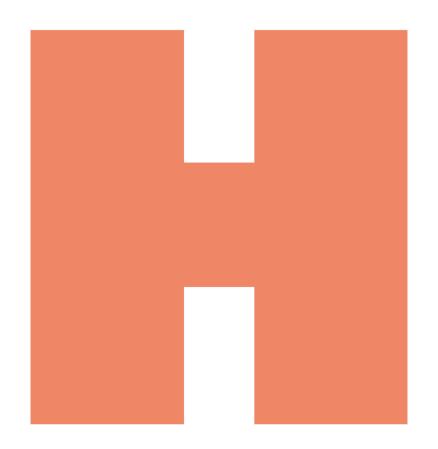
VALUES (concat ('JA', 'DEG'), 'Degroote', 'Jan', 'Mr.', 'Gentstraat 23', 'Gent', 'Oost-Vlaanderen', '9000', 'België', 095623147, null)

De constraints worden gecontroleerd...

Kolommen **niet vermeld** in 'insert', krijgen een **NULL** waarde, tenzij er een DEFAULT constraint bestaat, dan wordt de **DEFAULT** waarde toegekend.

Bij een **auto-increment** kolom worden waarden door het systeem gegeneerd bij het toevoegen van een rij. Deze kolom mag je dan ook **nooit** toevoegen aan een INSERT instructie.

Je kan de gegenereerde waarde wel achteraf opvragen.



Update.



UPDATE [LOW_PRIORITY] [IGNORE] table_reference
SET assignment_list
[WHERE where_condition]
[ORDER BY ...]
[LIMIT row_count]



- Wijzigen van alle rijen in een tabel
 - <u>Voorbeeld</u>: verhoog de prijs van alle producten met 10%

```
UPDATE Products
SET unitprice = (unitprice * 1.1)
```

- Wijzigen van 1 rij of een groep van rijen
 - Voorbeeld: verhoog de prijs van het product "Chocolade" met 10%

```
UPDATE products

SET unitprice = (unitprice * 1.1)

WHERE productname = 'Chocolade'
```

 Voorbeeld: verhoog de prijs van het product "Chocolade" met 10% en plaats 'aantal eenheden in voorraad' op 0

```
UPDATE products

SET unitprice = (unitprice * 1.1), unitsinstock = 0

WHERE productname = 'Chocolade'
```



 <u>Voorbeeld</u>: verhoog de eenheidsprijs van producten van de leveranciers met nummer SUP001, SUP002, SUP009, SUP022 met 10%.

UPDATE products

SET unitprice = (unitprice * 1.1)

WHERE suppliered IN (SUP001, SUP002, SUP009, SUP022)



CASE – NULLIF

<u>Voorbeeld</u>: Pas kolom reportsTo aan: indien reportsTo = 101, dan wordt de inhoud NULL

```
UPDATE employees

SET reportsTo =

CASE WHEN reportsTo = 101

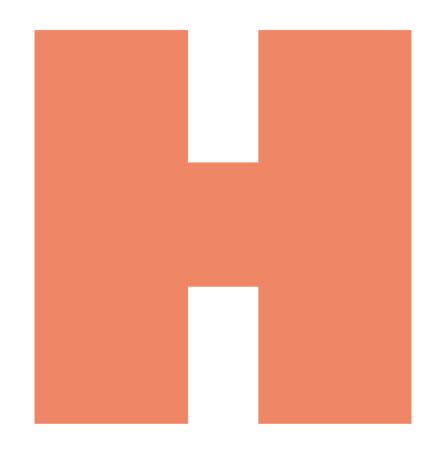
THEN NULL

END
```

```
UPDATE employees

SET reportsTo = NULLIF(reportsTo,101)
```





Delete.



Verwijderen van data -- DELETE

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE]
FROM tbl_name [PARTITION (partition_name [,partition_name] ...)]
[WHERE where_condition]
[ORDER BY ...]
[LIMIT row_count]
```



Verwijderen van data -- DELETE

- Verwijderen van rij(en)
 - Voorbeeld: verwijder de categorie 'Chocolade'

DELETE FROM categories

WHERE categoryName = 'Chocolade'

- Verwijderen van alle rijen in een tabel
 - via DELETE zal de auto increment soms gewoon verder lopen

DELETE FROM products

 via TRUNCATE zal de auto increment herstarten vanaf 1 (performanter: drop gevolgd door create)



Verwijderen van data -- DELETE

Voorbeeld: verwijder de gegevens van de opgegeven orders

DELETE FROM order

WHERE orderid IN (ORDER001, ORDER002, ORDER011, ORDER025)

Referentiële integriteitsregel:

- Bij het verwijderen (en updaten) van een aantal records in een tabel, kan het zijn dat records in een andere tabel ook verwijderd (aangepast) worden (vb. ON DELETE CASCADE).
- Het verwijderen van een record kan hierdoor soms falen

 bij het wissen van gegevens moeten de gegevens vaak trapsgewijs in de juiste volgorde in meerdere tabellen worden verwijderd. Dit wordt verder behandeld in 2TI.

Vergeet de WHERE clause niet (anders de volledige tabel leeg!)

