

© 2020 CODEMEMBERS - AUTEUR/ TOM VANHOUTTE

CODEMEMBERS

Cursus MYSQL

Inhoudstafel

Intro

In deze cursus zal manipulaties op een database zowel lokaal als online leren uitvoeren. De taal die we hiervoor zullen leren is mySQL. SQL = Structured Query Language.

LEGENDE	
	<u>Belangrijke info</u>
	<u>Notitie</u>
	<u>Oefening</u>
	<u>Opdracht</u>

De symbolen die u in de legende hiernaast ziet staan zullen we doorheen deze cursus gebruiken. Belangrijke info en oefeningen worden gezamenlijk aangeleerd en uitgelegd.
Notities kunnen door u als cursist worden toegevoegd.
Oeferingen volgen op de oefeningen en dient u individueel op te lossen om de opgedane kennis te toetsen.

Database

Een database is een georganiseerde verzameling van gegevens die toegankelijk is en wordt opgeslagen door een computersysteem. Databases worden ontwikkeld d.m.v. modelleertechnieken. Er is een verschil tussen relationele en niet-relationele databases. In deze cursus zullen we het hebben over relationele databases.

Voorbeelden van relationele databases zijn:
Mysql, Oracle, Access, Microsoft Sql Server, DB2, ...

Voordelen:

- snel opzoeken van informatie
- verschillende tabel die gerelateerd worden d.m.v. keys (sleutels)
- efficiënt opslaan van data
- rechtenstructuur voor gebruikers
- opvragen via één gestandardiseerde taal: SQL.

Installatie

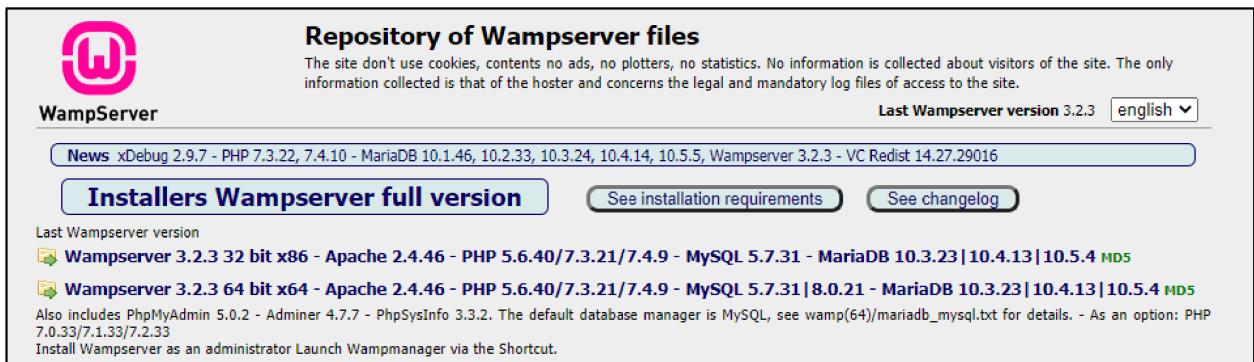
Een mySQL database wordt gebruikt voor webtoepassingen. D.w.z. dat we een werkende webserver dienen te hebben met een volledige installatie van mySQL op deze server. De server die we hiervoor zullen installeren is de Apache Webserver die de meest gebruikte webserver is op het internet. Daarnaast installeren we op deze webserver de mySQL omgeving. Er zijn verschillende mogelijkheden voor deze omgeving zoals: mongoDB, mariaDb, phpmyAdmin, ... Wij gebruiken phpmyAdmin, daar deze het meest wordt toegepast.

Windows - wampserver

Belangrijke info

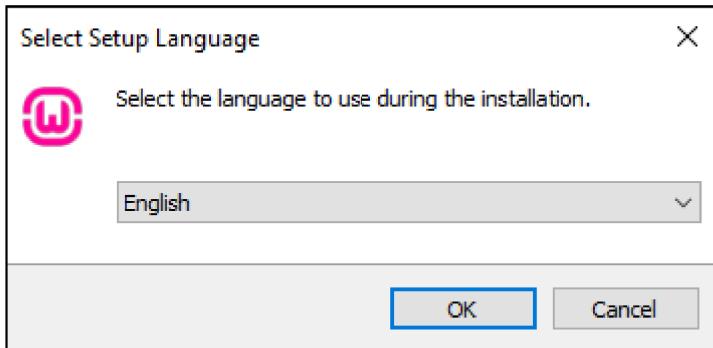
Om de apache webserver en de mysql omgeving te installeren surf je naar de volgende link:
<http://wampserver.aviatechno.net>

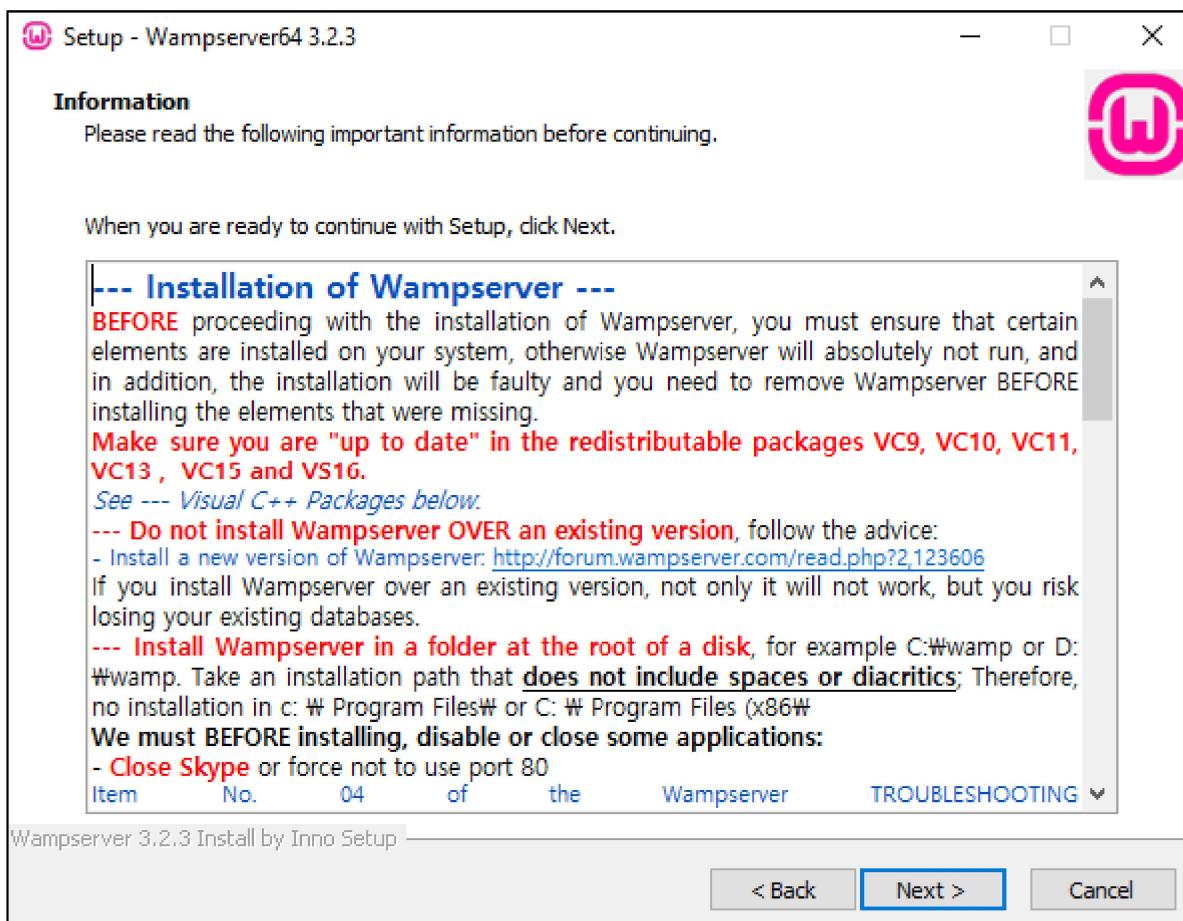
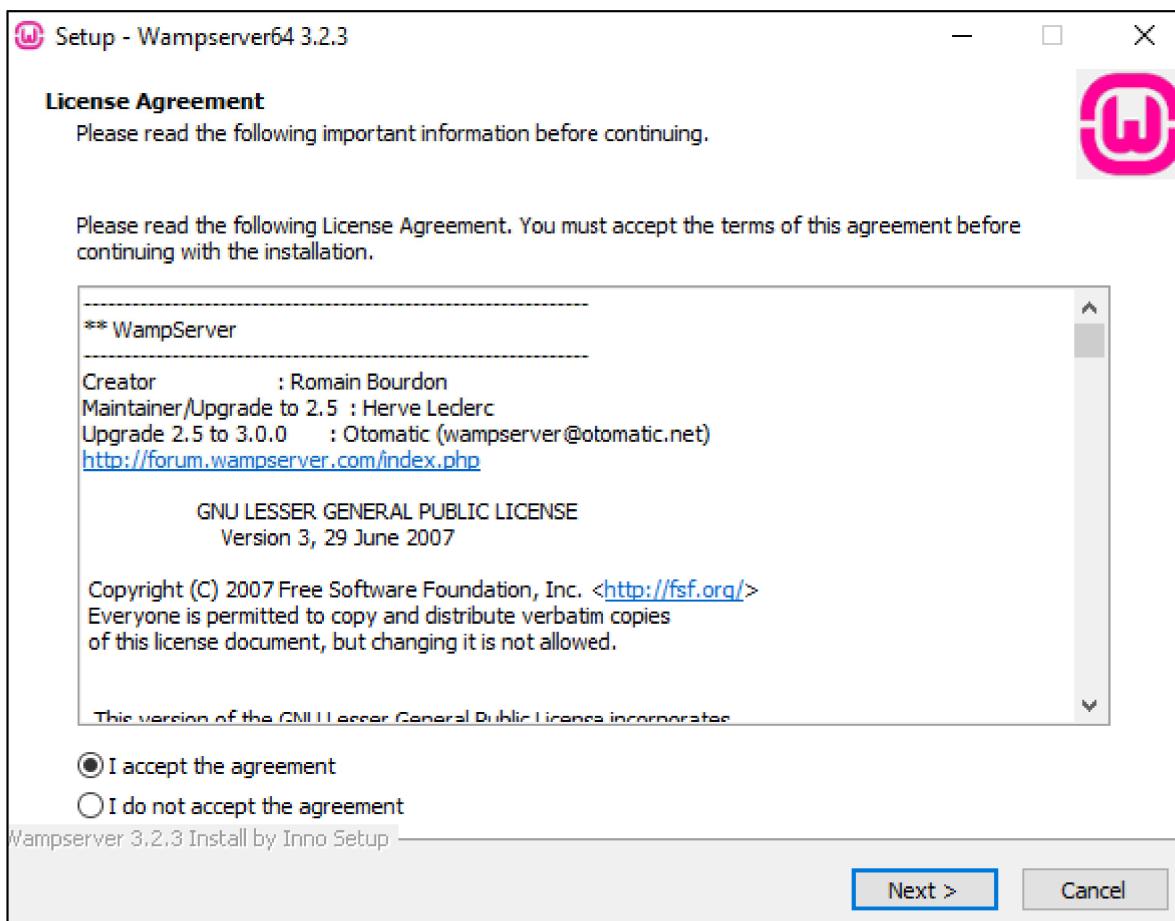
Download de gepaste versie (voor de meeste pc's zal dit de x64 versie zijn).

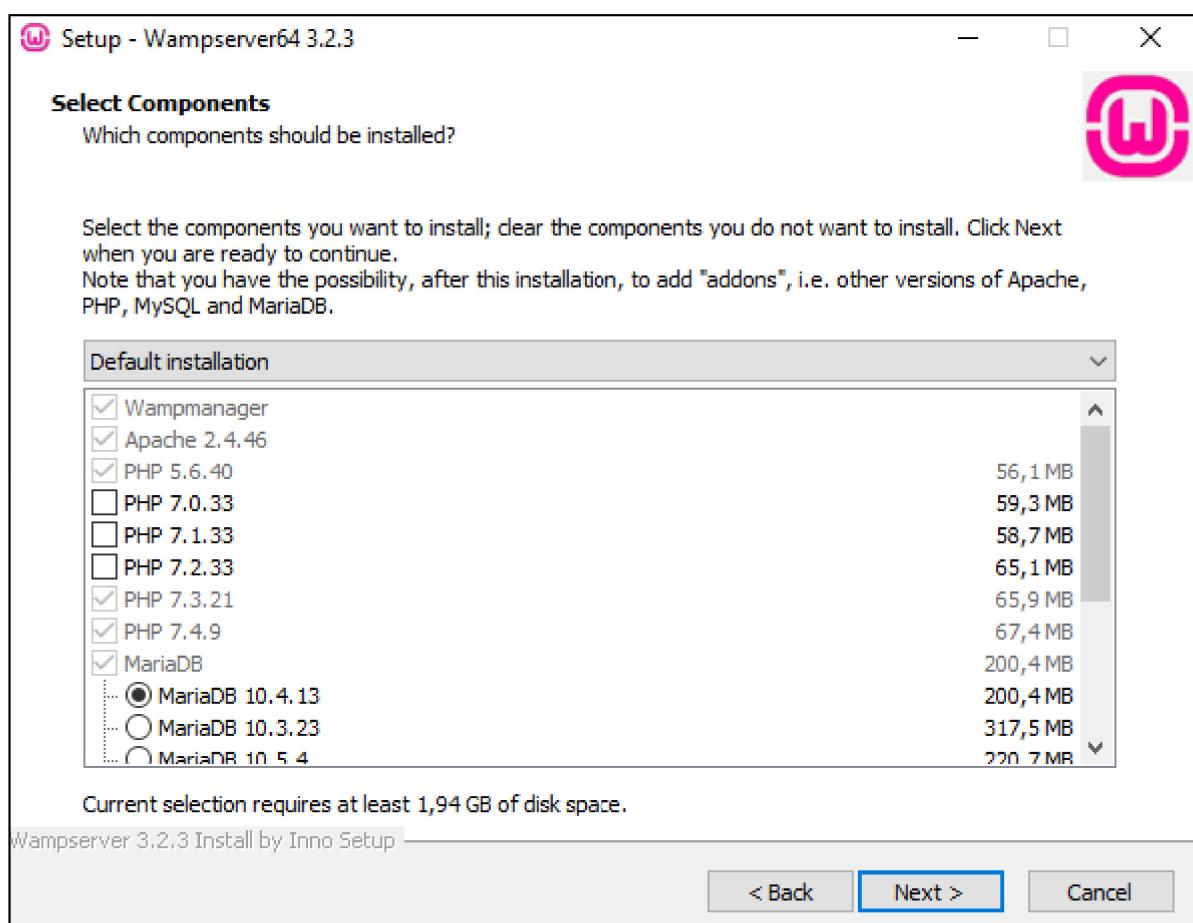
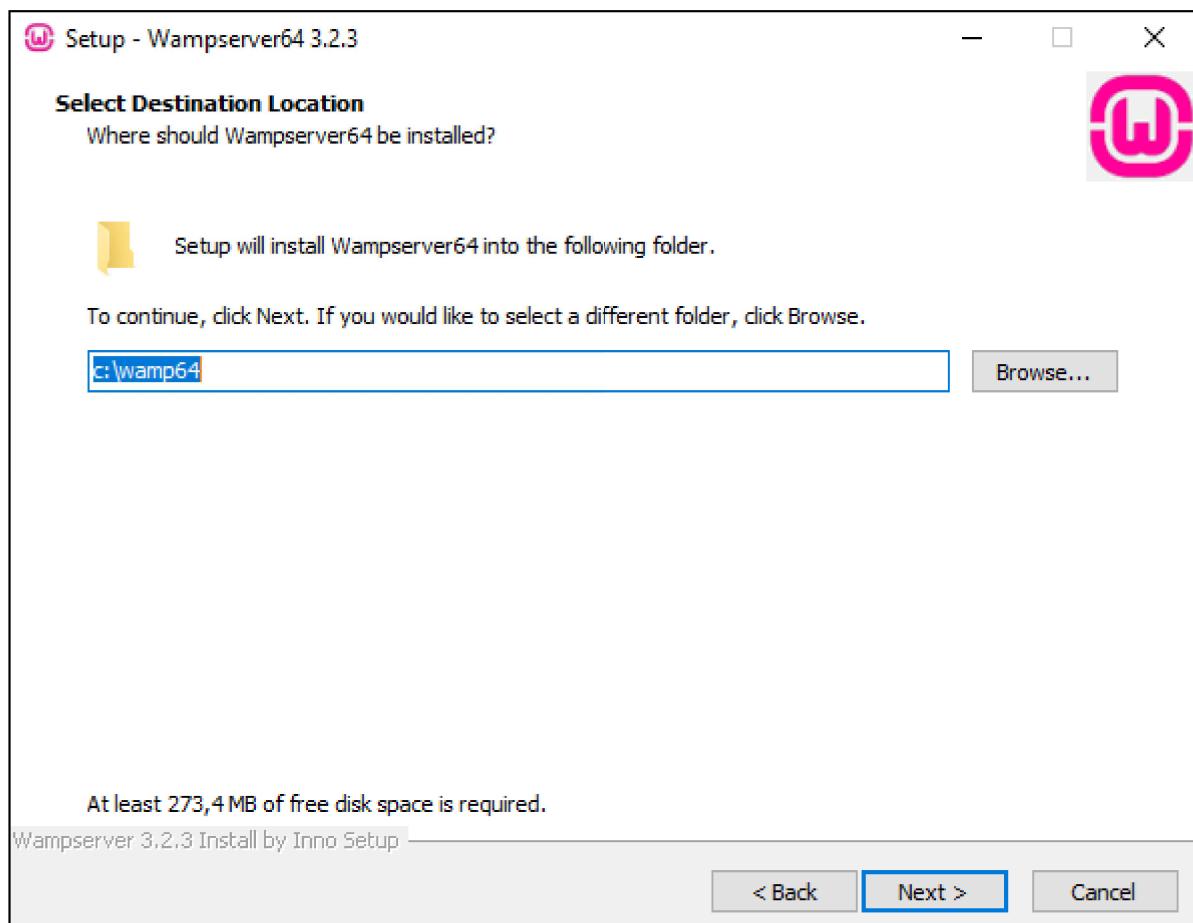


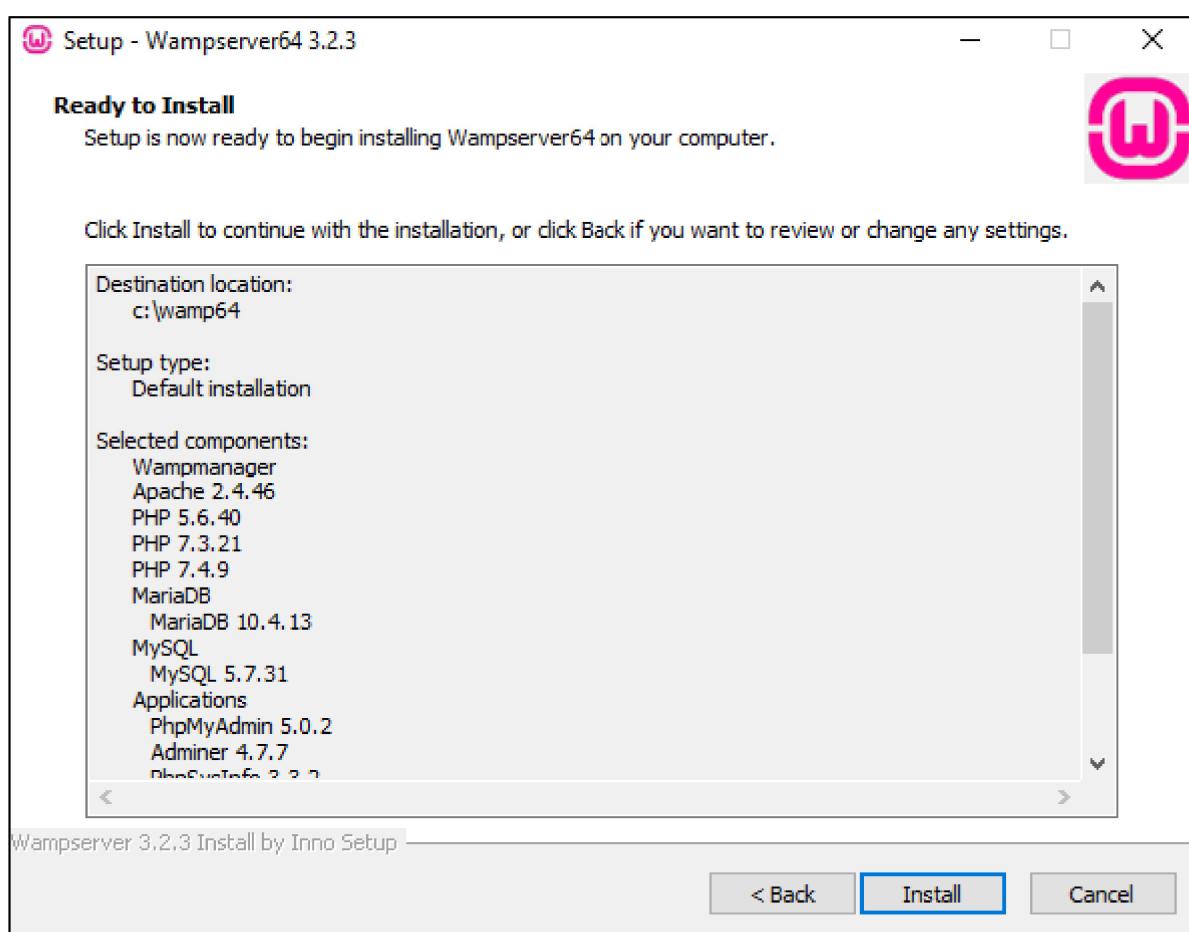
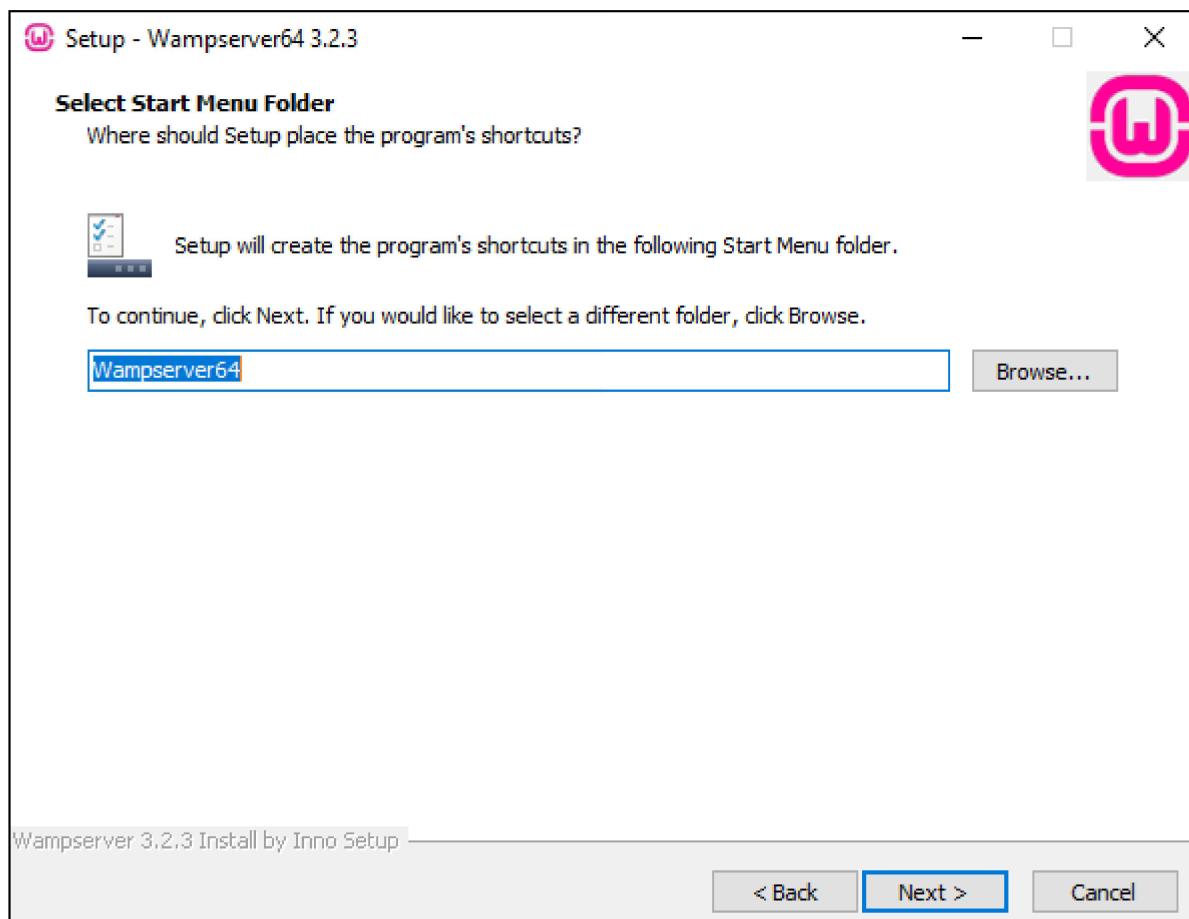
The screenshot shows the 'Repository of Wampserver files' page. At the top, there is a logo for 'WampServer' and a message stating: 'The site don't use cookies, contents no ads, no plotters, no statistics. No information is collected about visitors of the site. The only information collected is that of the hoster and concerns the legal and mandatory log files of access to the site.' Below this, it says 'Last Wampserver version 3.2.3' and 'english'. There are two main download links: 'Installers Wampserver full version' and 'See installation requirements' (in a blue button) and 'See changelog' (in a grey button). Below these, there are two more download links: 'Wampserver 3.2.3 32 bit x86 - Apache 2.4.46 - PHP 5.6.40/7.3.21/7.4.9 - MySQL 5.7.31 - MariaDB 10.3.23|10.4.13|10.5.4 MDS' and 'Wampserver 3.2.3 64 bit x64 - Apache 2.4.46 - PHP 5.6.40/7.3.21/7.4.9 - MySQL 5.7.31|8.0.21 - MariaDB 10.3.23|10.4.13|10.5.4 MDS'. A note below states: 'Also includes PhpMyAdmin 5.0.2 - Adminer 4.7.7 - PhpSysInfo 3.3.2. The default database manager is MySQL, see wamp(64)/mariadb_mysql.txt for details. - As an option: PHP 7.0.33/7.1.33/7.2.33' and 'Install Wampserver as an administrator Launch Wampmanager via the Shortcut.'

In de downloadsmap dubbelklik je op het uitvoeringsbestand en start je de installatie: volgt de schermafbeeldingen hieronder.









PHPmyadmin

De database omgeving waarin wij zullen werken is phpmyadmin. We zullen dus de MySQL taal rechtstreeks op verschillende databases uitvoeren. Bij manipulatie van een database kan je dus met deze taal acties uitvoeren die dikwijls niet omkeerbaar zijn. Bijvoorbeeld het verwijderen van een tabel in een database of het wijzigen van gebruikers, Heden ten dage draait alles om data die zeer waardevol is voor verschillende mogelijke partijen, waaronder personen met minder goede bedoelingen zoals hackers. Wanneer je bijvoorbeeld later een carrière als ethisch hacker wil uitoefenen, dan dien je dus de SQL-taal al zeer goed te kennen en de omgevingen waarin deze taal wordt uitgevoerd.

Openen



Na de installatie van de wampserver zal je dit icoontje terugvinden op je bureaublad.

Dubbelklik op dit icoontje op zowel de apache webserver te starten alsook MySQL. Je pc fungeert nu eigenlijk als een lokale webserver die NIET geconnecteerd is met het web. Een reëele webserver is WEL geconnecteerd met het web.

Webservers voorzien dus enerzijds hosting om je website of webapplicatie op te laden alsook de phpmyadmin databaseomgeving anderzijds.

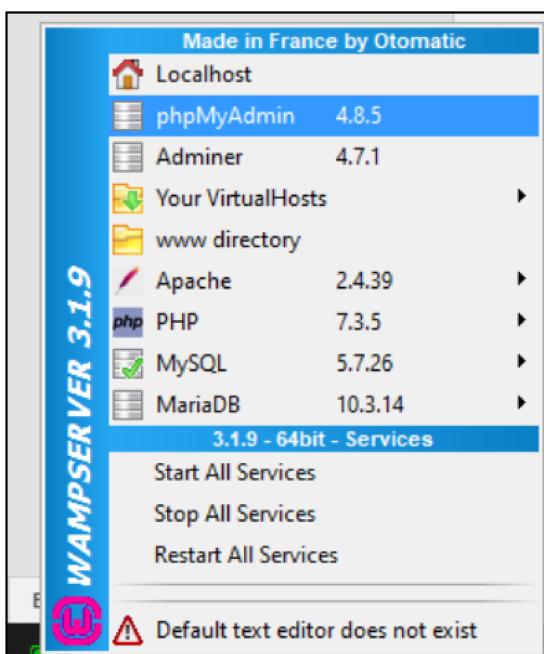
Als developer zullen we dus onze webapplicaties eerst lokaal maken alvorens deze op te laden naar een actieve hosting.



Wanneer je alles correct geïnstalleerd hebt en alles opgestart is, zul je rechts onderaan in je taakkalk hetzelfde icoontje zien staan die GROEN dient te zijn. Bij een groen icoontje werd alles correct gestart.

Bij een ORANJE icoontje werd ofwel de apache webserver of mysql niet goed opgestart.

Bij een ROOD icoontje werd niets opgestart.



Klik nu met je linker muisknop op dit GROENE icoontje en klik vervolgens op phpmyadmin om de database omgeving te openen. Deze omgeving zal jullie speelplaats worden tijdens deze cursus. Naargelang de evolutie van alle programma's kunnen de versies die in de schermafbeeldingen te zien zijn verschillen van deze cursus.

In het volgende scherm zal je nu kunnen inloggen. In de lokale ontwikkelomgeving dienen we geen speciale username of password te gebruiken. Standaard loggen we op phpmyadmin in met de user **root** en laten we password **blanco** staan. Klik vervolgens op **Go**.

Vanzelfsprekend wanneer we onze afgewerkte database zouden opladen naar een reëele webserver zal dit aangepast worden. Het zou bijzonder leuk zijn voor personen met slechte bedoelingen moest dit ook online op deze manier toegankelijk zijn.

Na het inloggen kom je terecht in de standaard omgeving van phpmyadmin. Hier kan je verschillende databases aanmaken en/of bevragen. Het belangrijkste in onderstaand scherm is de positionering nl. **Server:MySQL:3306**. d.w.z. dat je nu op het hoogste niveau van de MySQL Server bevindt.

CURSUS MYSQL

Er dient ook een woordje uitleg te worden gegeven over het onderstaande scherm. Hier zie je dat de **Web Server** die momenteel draaiende is dankzij wamp, de **Apache webserver** is.

De **Database Server** die we momenteel via phpmyAdmin bekijken is **mySQL**.

PHPmyAdmin is de huidige omgeving waarin **mySQL** momenteel draait.

Dit kan dus evengoed een andere omgeving zijn zoals mongoDB, mariaDB die eveneens mySQL draaien, maar worden hier niet besproken.

Database server

- Server: MySQL (127.0.0.1 via TCP/IP)
- Server type: MySQL
- Server connection: SSL is not being used ⓘ
- Server version: 5.7.26 - MySQL Community Server (GPL)
- Protocol version: 10
- User: root@localhost
- Server charset: UTF-8 Unicode (utf8)

Web server

- Apache/2.4.39 (Win64) PHP/7.3.5
- Database client version: libmysql - mysqlnd 5.0.12-dev - 20150407 - \$Id: 7cc7cc96e675f6d72e5cf0f267f48e167c2abb23 \$
- PHP extension: mysqli ⓘ curl ⓘ mbstring ⓘ
- PHP version: 7.3.5

phpMyAdmin

- Version information: 4.8.5, latest stable version: 4.9.5
- Documentation
- Official Homepage
- Contribute
- Get support
- List of changes
- License

Databases

Nieuw

Een nieuwe database creëren via phpMyAdmin is eenvoudig. Aan de linkerkant zie je een menu die alle databases zal bevatten. Je kan dus oneindig veel databases op phpMyAdmin draaien voor verschillende webapplicaties. Klik op **New** om een database aan te maken.

Databases

Zoals je hierboven kan zien zie je de characterset latin_swedish_ci. Dit is de standaard. Deze characterset bestaat uit 8 bits in maximale grootte. Standaard is dit meestal meer dan genoeg om tekens (letters, cijfers, speciale tekens) te bewaren.

In deze characterset zitten natuurlijk alle tekens die ook in de ASCII tabel zitten en meer. Waarom wordt dan de ASCII tabel niet gebruikt? ASCII telde initieel 127 symbolen die konden bewaard worden in een 7-bits karakterset. In een 8-bits karakterset kunnen er dus meer symbolen worden gebruikt.

Voor websites zitten we tegenwoordig aan 16-bits in grootte. De characterset die daar zal worden gebruikt is de huidige standaard UTF-8.

In de meeste gevallen heb je echter genoeg met de swedisch karakterset. Wil je zeker zijn dan kan je de characterset hier ook aanpassen naar UTF-8.

Kies nu **utf8_general_ci** als characterset en druk **create**.

De database werd nu aangemaakt. De engine die wordt gebruikt om de database te bevragen in phpMyAdmin is standaard MyISAM. Deze heeft als nadeel dat we geen relaties kunnen leggen tussen tabellen in deze omgeving. We dienen deze om te zetten naar de andere engine van phpMyAdmin, nl. InnoDB

Engine InnoDB

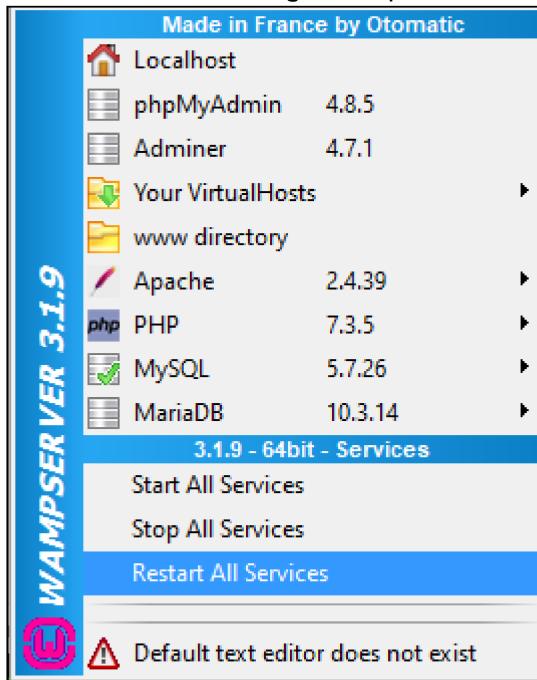
Om de engine te wijzigen dienen we het bestand my.ini te openen op onderstaande locatie (zie schermafbeelding).

This PC > Windows (C:) > wamp64 > bin > mysql > mysql5.7.26				
	Name	Date modified	Type	Size
	bin	9/28/2019 2:49 PM	File folder	
	data	9/23/2020 9:03 PM	File folder	
	docs	9/28/2019 2:49 PM	File folder	
	include	9/28/2019 2:49 PM	File folder	
	lib	9/28/2019 2:49 PM	File folder	
	share	9/28/2019 2:49 PM	File folder	
	COPYING	4/13/2019 3:32 PM	File	18 KB
	my.ini	9/23/2020 9:03 PM	Configuration sett...	7 KB
	README	4/13/2019 3:32 PM	File	3 KB
	wampserver.conf	12/11/2015 11:55 AM	CONF File	1 KB

In my.ini zoek je achter de myISAM engine en wijzig je deze naar InnoDB.

```
; The default storage engine that will be used when create new tables
default-storage-engine=InnoDB
; New for MySQL 5.6 default_tmp_storage_engine if skip-innodb enable
; default_tmp_storage_engine=MYISAM
```

Herstart nu de volledige wampserver als volgt om de engine in te laden:



Verwijderen

Databases kunnen we ook verwijderen uit phpMyAdmin.

- klik bovenaan op **Server:MySQL:3306**
- klik vervolgens op het tabblad **Databases**
 - wagens**
 - Selecteer de te verwijderen database
 - Klik op **Drop** en druk **OK**

Sample Databases

Voor mySQL werden er enkel databases aangemaakt speciaal voor studenten die de taal wensen onder de knie te krijgen.

Link:<https://dev.mysql.com/doc/index-other.html>

De databases die o.a. worden gebruikt zijn employee, world, sakila en menagerie.

Wij zullen de **sakila** database gebruiken, maar eerst installeren. **Download** hier het **Zip** bestand van de database. We pakken deze uit op de c:\ schijf.

Example Databases

Title	Download DB	HTML Setup Guide	PDF Setup Guide
employee data (large dataset, includes data and test/verification suite)	GitHub	View	US Ltr A4
world database	Gzip Zip	View	US Ltr A4
world_x database	TGZ Zip	View	US Ltr A4
sakila database	TGZ Zip	View	US Ltr A4
menagerie database	TGZ Zip		

Installatie SAKILA database

Hiervoor zullen we de **HTML Setup Guide** gebruiken van de sakila database. Druk dan op **Installation**.

Sakila Sample Database

Table of Contents

- [1 Preface and Legal Notices](#)
- [2 Introduction](#)
- [3 History](#)
- [4 Installation](#)
- [5 Structure](#)
- [6 Usage Examples](#)
- [7 Known Issues](#)
- [8 Acknowledgments](#)
- [9 License for the Sakila Sample Database](#)
- [10 Note for Authors](#)
- [11 Sakila Change History](#)

CURSUS MYSQL

In de volgende pagina zie je nu enkele commando's die we in een console dienen in te geven:

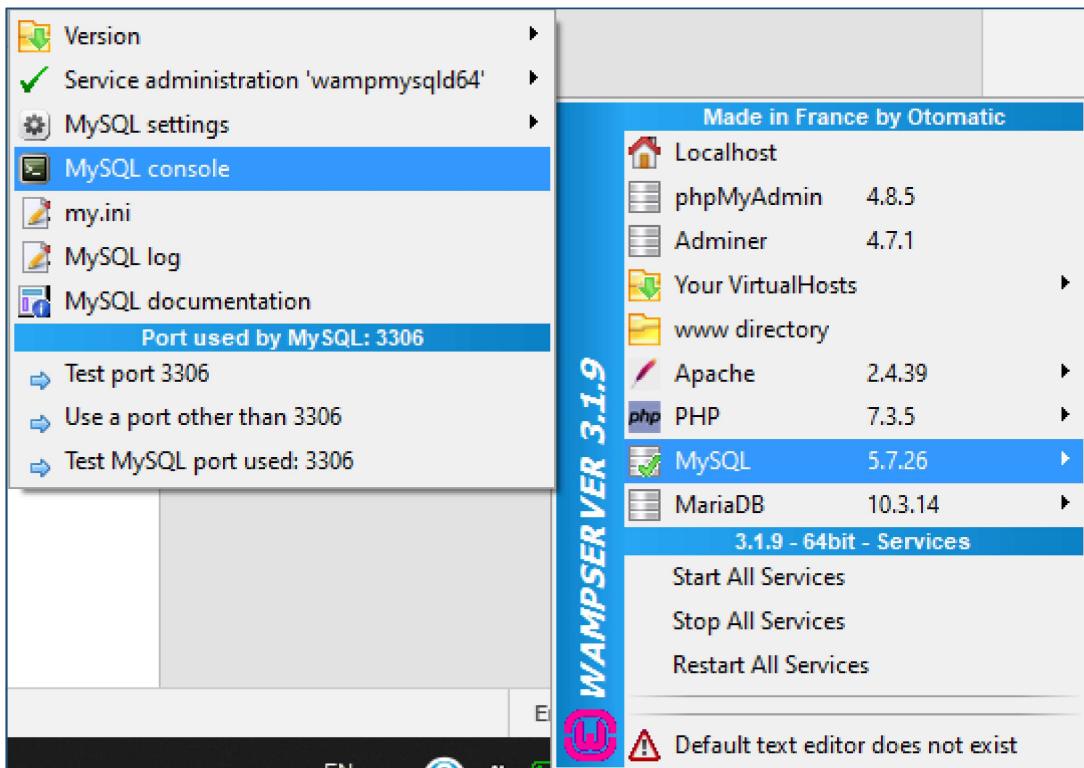
To install the Sakila sample database, follow these steps:

1. Extract the installation archive to a temporary location such as `c:\temp\` or `/tmp/`. When you unpack the `sakila-schema.sql` and `sakila-data.sql` files.

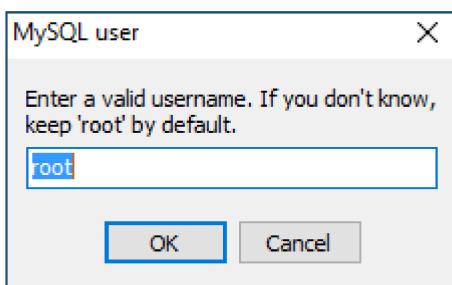
2. Connect to the MySQL server using the `mysql` command-line client with the following command:

```
1 | shell> mysql -u root -p
```

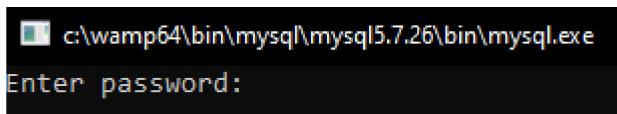
De console die we hiervoor gaan gebruiken is de MySQL console die je in de schermafbeelding hieronder ziet staan.



De standaard username van phpMyAdmin zal worden gevraagd. Druk hier op **OK**.



Het paswoord die je dient in te vullen is **blanco**. Druk dus gewoon op **ENTER**.



CURSUS MYSQL

Wanneer alles correct werd uitgevoerd zit je in de console van MySQL.

```
c:\wamp64\bin\mysql\mysql5.7.26\bin\mysql.exe
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 58
Server version: 5.7.26 MySQL Community Server (GPL)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

Nu dienen we het de sakila database die we uitgepakt hebben op de c schijf te installeren.
Eerst wordt de structuur (tabellen) geïnstalleerd met het volgende commando. Let op dat de locatie juist is.

```
mysql> SOURCE C:/sakila-db/sakila-schema.sql
```

Vervolgens gaan we de data ook in de tabellen injecteren met het volgende commando.

```
mysql> SOURCE C:/sakila-db/sakila-data.sql
```

Opdracht

Probeer ook de world database te installeren.

CURSUS MYSQL

Sluit nu de consoletoepassing af en open terug phpMyAdmin.

Resultaat:

The screenshot shows the phpMyAdmin interface for the MySQL 3.306 server. The database selected is 'sakila'. The top navigation bar includes tabs for Structure, SQL, Search, Query, Export, Import, Operations, Privileges, Routines, Events, and Triggers. Below the navigation is a 'Filters' section with a search input field. The main content area displays a table of database tables, each with a checkbox, an icon, and a set of actions (Browse, Structure, Search, Insert, Empty, Drop). The table includes columns for Row count, Type, Collation, Size, and Overhead. The 'country' table is currently selected, indicated by a blue background.

Table	Action	Rows	Type	Collation	Size	Overhead
actor	Browse Structure Search Insert Empty Drop	200	InnoDB	utf8mb4_general_ci	32 Kib	-
actor_info	Browse Structure Search Insert Empty Drop	~0	View	---	-	-
address	Browse Structure Search Insert Empty Drop	603	InnoDB	utf8mb4_general_ci	112 Kib	-
category	Browse Structure Search Insert Empty Drop	16	InnoDB	utf8mb4_general_ci	16 Kib	-
city	Browse Structure Search Insert Empty Drop	600	InnoDB	utf8mb4_general_ci	64 Kib	-
country	Browse Structure Search Insert Empty Drop	109	InnoDB	utf8mb4_general_ci	16 Kib	-
customer	Browse Structure Search Insert Empty Drop	599	InnoDB	utf8mb4_general_ci	128 Kib	-
customer_list	Browse Structure Search Insert Empty Drop	~0	View	---	-	-
film	Browse Structure Search Insert Empty Drop	1,000	InnoDB	utf8mb4_general_ci	272 Kib	-
film_actor	Browse Structure Search Insert Empty Drop	5,462	InnoDB	utf8mb4_general_ci	272 Kib	-
film_category	Browse Structure Search Insert Empty Drop	1,000	InnoDB	utf8mb4_general_ci	80 Kib	-
film_list	Browse Structure Search Insert Empty Drop	~0	View	---	-	-
film_text	Browse Structure Search Insert Empty Drop	1,000	InnoDB	utf8mb4_general_ci	192 Kib	-
inventory	Browse Structure Search Insert Empty Drop	4,581	InnoDB	utf8mb4_general_ci	368 Kib	-
language	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_general_ci	16 Kib	-
nicer_but_slower_film_list	Browse Structure Search Insert Empty Drop	~0	View	---	-	-
payment	Browse Structure Search Insert Empty Drop	16,049	InnoDB	utf8mb4_general_ci	2.1 MiB	-
rental	Browse Structure Search Insert Empty Drop	16,044	InnoDB	utf8mb4_general_ci	80 Kib	-
sales_by_film_category	Browse Structure Search Insert Empty Drop	~0	View	---	-	-
sales_by_store	Browse Structure Search Insert Empty Drop	~0	View	---	-	-
staff	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	48 Kib	-
staff_list	Browse Structure Search Insert Empty Drop	~0	View	---	-	-
store	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	48 Kib	-

Tabellen

Een database bestaat uit genormaliseerde tabellen. Normalisatie is een onderwerp die we later zullen bekijken. Het komt er op neer dat iedere tabel zijn eigen specifieke data heeft en gerelateerd kan zijn met andere tabellen in de database.

Voorbeeld:

De tabel met gebruikers kan gerelateerd zijn met de tabel van adressen.

Een gebruiker kan dus één of meerdere adressen hebben.

Datatypes

Tabellen bestaan uit velden. Ieder veld heeft zijn eigen veldgebonden data en zijn datatype. Een huisnummer zal bijvoorbeeld van het type integer zijn.

De mogelijke datatypes die we kunnen gebruiken in MySQL zijn:

Data Type=INTEGERS	grootte	unsigned
INT	-2.1 biljoen tot 2.1 biljoen	0 tot 4.2 biljoen
TINYINT	-128 tot 127	0 tot 255
SMALLINT	-32768 tot 32767	0 tot 65535
MEDIUMINT	-8.3 miljoen tot 8.3 miljoen	0 tot 16.7 miljoen
BIGINT	-2 ⁶³ tot -2 ⁶³⁻¹	0 tot 2 ⁶⁴⁻¹

Data Type=FLOATING POINT	gebruik
FLOAT	Decimalen (single precisie)
DOUBLE	Grote Decimalen (double precisie)
DECIMAL	Waarden waar afrondingerrors niet worden aanvaard. Bijv. geld, 65 digits voor de komma en 30 na de komma precisie. Dit wordt gebruikt om o.a. geld te berekenen.

Voorbeeld van gebruik van FLOATING POINT:

De gebruiker tik respectievelijk in:

Rij 1:

0.6 ,0.6 ,0.6

Rij 2:

0.00006, 0.00006, 0.00006

FLOAT	DOUBLE	DECIMAL
0.6	0.6	0.60000
0.00006	0.00006	0.00006

Wanneer we nu met deze getallen rekenen dan krijgen we het volgende resultaat:

we vermenigvuldigen alles met 1000

FLOAT	DOUBLE	DECIMAL
600.0000238418579	600	600.00000
0.05999999848427251	0.06000000000000005	0.06000

Conclusie: Decimals zijn het meest accuraat te gebruiken.

Data Type= STRINGS en BINAIRE DATA	gebruik
VARCHAR	Korte, variable lengte van tekst
CHAR	Korte,vaste lengte van tekst,bijv. Encrypted passwords
TEXT	Lange tekst, bijv. artikelen
ENUM	1 waarde van een voorgedefinieerde lijst. Voorbeeld van lijst: zomer, herfst, lente, winter
BLOB	Opslag van beelden en audiobestanden als gecomprimeerde files.

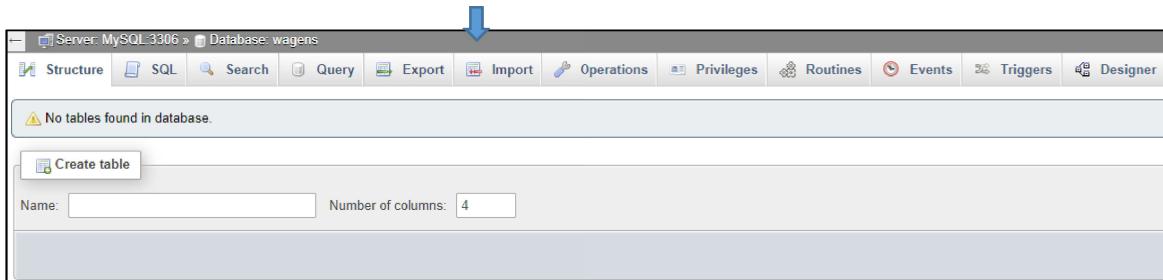
Data Type= DATUM EN TIJD	format
DATE	YYYY-MM-DD
TIME	hh:mm:ss
DATETIME	YYYY-MM-DD hh:mm:ss
TIMESTAMP	YYYY-MM-DD hh:mm:ss
JAAR	YYYY

De meeste gebruikte datatypes zijn:

Data Type	GEBRUIK
INT	Integers
TINYINT	Kleine Integers zoals leeftijd
DECIMAL	Geld en afmetingen
VARCHAR	Korte tekst
TEXT	Lange tekst
DATETIME	Datum

Nieuw

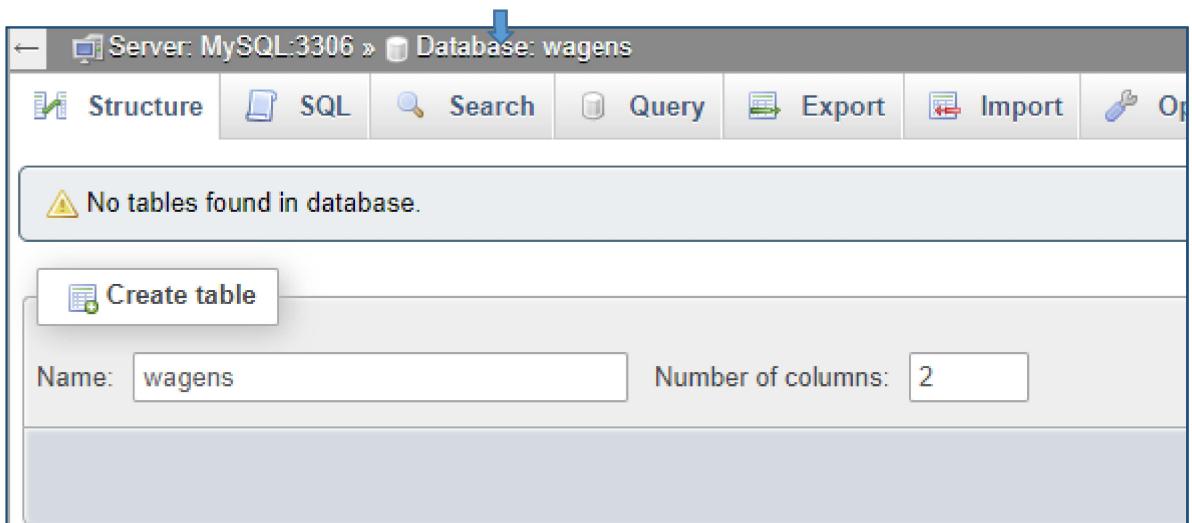
Een nieuwe tabel aanmaken is eenvoudig. Je dient eerst je database te selecteren. Je kan steeds controleren of je in de correcte database zit bovenaan phpMyAdmin. Net zoals in windows verkenner is dit een folder structuur die vertrekt van het server niveau.



In ons voorbeeld hebben we het over wagens. Wagens bestaan uit merken en modellen. Hiervoor creëren we dus 2 tabellen.

De eerste tabel wagens zal 2 velden bevatten: id, merk.

Vul de **naam** van de tabel in en duid hiervoor **2** kolommen aan voor het aantal velden. Klik dan op **Go**.



Conventie bij het schrijven van veldnamen: ALLEMAAL KLEINE LETTERS! GEEN CAMELCASE zoals variabelen in PROGRAMMEERTALEN.

Vul in zoals onderstaande afbeelding en klik op **SAVE**. Je hebt nu een lege tabel zonder data aangemaakt in de database van wagens.

The screenshot shows the 'Structure' tab for the 'wagens' table. It has two columns: 'id' (INT, 11, None, UNSIGNED, PRIMARY) and 'merk' (VARCHAR, 255, None). The 'Storage Engine' is set to InnoDB. At the bottom right, there are 'Preview SQL' and 'Save' buttons.

Name	Type	Length/Values	Default	Collation	Attributes	Null Index	A_I	Comments	Virtuality	Move column
id	INT	11	None		UNSIGNED	<input checked="" type="checkbox"/>	PRIMARY	PRIMARY		
merk	VARCHAR	255	None							

Veld instellingen

Hier zullen we kort alle mogelijke instellingen bespreken tijdens het maken van velden in een tabel.

Name

Hier worden de kolomkoppen of veldnamen ingegeven. De eerste **veldnaam** die wij bijna altijd zullen invullen is het **id** van de tabel. Het **id** is ook meestal een oplopende nummering die start van 1 met een **auto_increment**. Auto_increment zorgt dus m.a.w. automatisch dat een volgend **record** automatisch verder wordt genummerd. Dit id is **UNIEK** en wordt de **PRIMARY KEY** van een tabel genoemd.

Een **record** is één enkele data-lijn in de database met ingevulde data per veld.

Type

Het type per veld is het datatype die we dienen te bepalen. Aangezien een id een oplopende nummering is zal dit van het type integer zijn. Hier kiezen we **INT**.

Een VARCHAR kan bijvoorbeeld gebruikt worden voor een string, DATETIME voor een datum, ...

Length/values

Aangezien het datatype bijvoorbeeld INT is kunnen we tot ca 2 biljoen in cijfers laten oplopen.

Dankzij lengte beperken we het aantal karakters toch. Veelal wordt er bij een id 11 karakters aan lengte toegevoegd die voor de meeste database meer dan voldoende is.

Het laatste id zou dus in principe 99999999999 kunnen zijn.

Default values

Hiermee kan je een standaard waarde meegeven wanneer een veld niet zou worden ingevuld door een gebruiker.

Er zijn 4 opties:

none = is default en blijft dus leeg

NULL = hier wordt een nullable in het veld geschreven. Je zal dan het woord NULL zien staan in het veld.

As defined: hier verschijnt een extra veld waar je zelf een default waarde kan schrijven. Deze default waarde wordt in het veld van de database weggeschreven wanneer de gebruiker niets zou hebben geschreven.

Current_time_stamp = Zal de datumtijdstempel wegschrijven van de pc.

Collation

Voor ieder veld kan je een aparte character set toevoegen indien je dit nodig zou hebben. In de meeste gevallen is de initiële character set van de database voldoende en blijft dit veld leeg.

Attributes

BINARY

De opslag van een ingetikte waarde zal binair gebeuren (1 en 0)

UNSIGNED

Een unsigned value kan enkel positieve getallen opslaan in de database.

Zie tabel datatypes (voorgaand).

UNSIGNED ZERO FILL

Zero fill zorgt ervoor dat alle ontbrekende characters met een 0 worden opgevuld.

Bijvoorbeeld: id(11)

id(11) kan een lengte van 11 getallen opslaan. Wanneer we nu het getal 1 opslaan dan wordt dit met zero fill: 00000000001

on update CURRENT_TIMESTAMP

Wanneer er een veld van het datatype datetime wordt gebruikt dan kunnen we hier ervoor zorgen dat automatisch de huidige datumtijdstempel van de pc wordt gebruikt wanneer een gebruiker een wijziging aanbrengt op dit record.

Null

Het veld mag leeg zijn wanneer aangevinkt.

Index

PRIMARY

Met deze optie kan je een veld de primaire sleutel maken van je tabel (PK = Primary KEY)

In combinatie met A-I (auto increment), maak je een primaire sleutel dus aan die daarnaast ook uniek is

UNIQUE

Wanneer je een veld UNIQUE zet, dan zal je dubbele waarden vermijden in een database. Dit veld kan wel een NULL value bevatten.

INDEX

MySQL gebruikt indexen om snel rijen met specifieke kolomwaarden te vinden. Zonder index moet MySQL de hele tabel scannen om de relevante rijen te lokaliseren. Hoe groter de tafel, hoe langzamer hij zoekt.

A_I

Wanneer aangevinkt zorgt A_I voor een oplopende nummering van een veld met een datatype integer.

COMMENTS

Hier kan je als developer een beschrijving geven van de bedoeling van een veld.

Code: CREATE TABLE

Een tabel kan je ook met code aanmaken. We maken dus hier gebruik van de MySQL-taal.

 Oefening

Voorbeeld:

```
CREATE TABLE Klanten (
    id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    voornaam VARCHAR(30) NOT NULL,
    familienaam VARCHAR(30) NOT NULL,
    email VARCHAR(50),
    reg_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
)
```

Na het gegevenstype kunt u voor elke kolom andere optionele attributen specificeren:

- NOT NULL - Elke rij moet een waarde voor die kolom bevatten, null-waarden zijn niet toegestaan
- DEFAULT-waarde - Stel een standaardwaarde in die wordt toegevoegd als er geen andere waarde wordt doorgegeven
- UNSIGNED - Gebruikt voor number datatypes , beperkt de opgeslagen gegevens tot positieve getallen en nul
- AUTO INCREMENT - MySQL verhoogt automatisch de waarde van het veld met 1 telkens wanneer een nieuw record wordt toegevoegd
- PRIMARY KEY - Wordt gebruikt om de rijen in een tabel uniek te identificeren. De kolom met PRIMARY KEY-instelling is vaak een ID-nummer en wordt vaak gebruikt met AUTO_INCREMENT

Opdracht

Maak een nieuwe tabel **modellen** aan in de tabel wagens. Zorg dat er 3 velden aanwezig zijn. Het eerste veld **id** is een integer die positieve primaire sleutel is met autonummering en een grootte van 11 heeft. Het tweede veld is **merkid** en is een positieve integer met een grootte van 11 en kan geen null bevatten. Het derde veld is naam en zal alle modelnamen bevatten. Dit is een string met een grootte van 255 en kan geen null bevatten.

Oplossing:

```
CREATE TABLE merken (
    id INT(11) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    merkid INT(11) UNSIGNED NOT NULL,
    naam VARCHAR(255) NOT NULL
)
```

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)		UNSIGNED	No	None		AUTO_INCREMENT	Change Drop More
2	merkid	int(11)		UNSIGNED	No	None			Change Drop More
3	naam	varchar(255)	utf8_general_ci		No	None			Change Drop More

Momenteel hebben we dus een database met de naam wagens. In deze database hebben we 2 tabellen, nl. merken en modellen. Wat we nog NIET hebben is een relatie tussen deze 2 tabellen.

Relaties

We sommen eerst de meest voorkomende relaties op die mogelijk zijn in een database tussen 2 of meerdere tabellen.

- één-op-één
- één-op-veel
- veel-op-veel

Relatie: één-op-veel

In ons voorbeeld hebben we bijvoorbeeld een model: AUDI
Audi heeft meerdere modellen in zijn gamma: A3, A4, A5,...

Dit noemen we een één-op-veel-relatie, want het merk audi (één) heeft meerdere modellen (veel).

Een relatie dienen we dus ook te definiëren. Momenteel hebben we reeds alle velden in de tabellen gedefinieerd, maar nog niet de relatie.

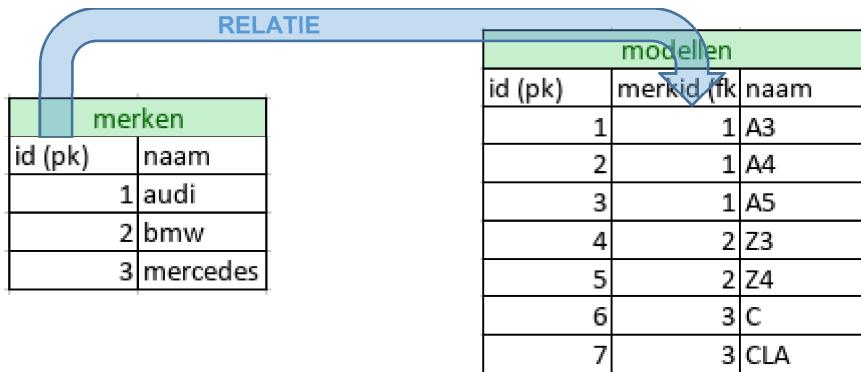
CURSUS MYSQL

Wanneer we de tabellen even schematisch bekijken in de designer tab op het niveau van de database dan kan je dit zien. Er is duidelijk geen link dus deze 2 tabellen. Deze lijken te ‘zweven’ naast elkaar

wagens merken
id : int(11) unsigned
wagens : varchar(255)

wagens modellen
id : int(11) unsigned
merkid : int(11) unsigned
naam : varchar(255)

Hieronder zie je wat de bedoeling zou moeten zijn in een **één-op-veel** relatie datagewijs.



Zoals je kan zien hebben beide tabellen een unieke primary key. Het merkid echter is een sleutel die vreemd is aan de tabel modellen en die werd geërfd van de tabel merken. Deze sleutel noemen we een FOREIGN KEY.

Om nu de relatie te bewerkstelligen in phpMyAdmin dienen we de TWEEDEN tabel modellen te selecteren en het tabblad STRUCTURE aan te klikken. Klik vervolgens op RELATION VIEW.

The screenshot shows the 'Relation view' tab selected in the phpMyAdmin interface for the 'modellen' table. It displays the 'Foreign key constraints' section where a constraint named 'model_merken' is being configured. The configuration shows 'ON DELETE RESTRICT' and 'ON UPDATE RESTRICT'. The 'Column' dropdowns show 'merkid' and 'wagens' respectively, and the 'Table' dropdown shows 'merken' with 'id' selected as the foreign key column.

De opties die mogelijk zijn bij “on delete en on update” zijn:

RESTRICT

Wanneer je data die verbonden is in 2 tabellen zou willen wissen, dan zal dit niet lukken. Wanneer je bijvoorbeeld audi in de tabel merken zou willen zal de database een restriction error geven omdat je modellen van audi in de tabel modellen zitten hebt.

CASCADE

Cascade zal wel wissen. Wanneer je audi zou wissen in de tabel merken dan zullen al zijn modellen in de tabel modellen ook gewist worden.

SET NULL

Wanneer je audi zou wissen in de tabel merken dan zullen de rijen in de tabel modellen niet gewist worden maar zal de inhoud van de rij op NULL worden geplaatst.

Wanneer we nu kijken naar de tabel modellen dan zie je een grijze sleutel staan die de foreign key weergeeft. Een gouden sleutel is de primary key van de tabel zelf.

#	Name	Type	Collation	Attributes	Null	Def
1	id	int(11)		UNSIGNED	No	None
2	merkid	int(11)		UNSIGNED	No	None
3	naam	varchar(255)	utf8_general_ci		No	None

Op het database niveau kan je terug naar het designer tabblad gaan en zie je het uiteindelijke resultaat van een één-op-veel relatie.



Code: FOREIGN KEY CONSTRAINTS

Wanneer je alle bovenstaande acties in code zou willen uitvoeren dan kan dit natuurlijk ook. WIS de tabel modellen uit de database.

Open nu het SQL tabblad op database niveau en voeg onderstaande code toe:

```

CREATE TABLE modellen (
    id INT(11) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    merkid INT(11) UNSIGNED NOT NULL,
    naam VARCHAR(255) NOT NULL,
    FOREIGN KEY (merkid) REFERENCES merken(id)
        ON UPDATE RESTRICT ON DELETE RESTRICT
)
    
```

Bovenstaande geeft hetzelfde resultaat terug en maakt de tabel modellen aan met de foreign key en zijn restriction.

CRUD

CRUD = CREATE READ UPDATE DELETE

Wanneer we over CRUD spreken, spreken we over alle mogelijk manipulaties die we op één of meerdere tabellen van een database kunnen uitvoeren.

De SQL syntax is de taal die we nodig zullen hebben om deze handelingen/manipulaties uit te voeren.

SELECT STATEMENT

We starten eerst met het READ gedeelte van de CRUD. Deze wordt in SQL uitgevoerd door het SELECT STATEMENT

Hieronder zie je de voorstelling van het volledige select statement die gebruikt KAN worden.

`SELECT field_references`

```
[ALL | DISTINCT | DISTINCTROW ]
[HIGH_PRIORITY]
[STRAIGHT_JOIN]
[SQL_SMALL_RESULT] [SQL_BIG_RESULT] [SQL_BUFFER_RESULT]
[SQL_CACHE | SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS]
select_expr [, select_expr] ...
[into_option]
[FROM table_references
  [PARTITION partition_list]]
[WHERE where_condition]
[GROUP BY {col_name | expr | position}
  [ASC | DESC], ... [WITH ROLLUP]]
[HAVING where_condition]
[ORDER BY {col_name | expr | position}
  [ASC | DESC], ...]
[LIMIT {[offset,] row_count | row_count OFFSET offset}]
```

We hebben de SAKILA database samen geïnstalleerd. Deze gaan we nu gebruiken om het select statement in detail aan te leren.

OEFENINGEN

BASIS SELECT EN/OF ORDER BY

Belangrijke info

ORDER BY: zorgt voor het alfabetisch sorteren van velden. Opties zijn ASC en DESC
DISTINCT: zorgt dat een VOLLEDIG record (rij) UNIEK is in het resultaat.

- Selecteer alle velden uit de tabel actor

```
SELECT *
```

```
FROM actor
```

- Selecteer enkel familienaam en voornaam uit de tabel actor

```
SELECT actor.last_name, actor.first_name
```

```
FROM actor
```

- Selecteer enkel familienaam en voornaam uit de tabel actor en sorteer deze eerst volgens familienaam (A-Z) en vervolgens volgens voornaam (A-Z)

```
SELECT actor.last_name, actor.first_name
```

```
FROM actor
```

```
ORDER BY actor.last_name ASC, actor.first_name DESC
```

- Selecteer enkel de unieke familienaam en voornaam uit de tabel actor en sorteert Z-A

```
SELECT DISTINCT actor.last_name, actor.first_name
```

```
FROM actor
```

```
ORDER BY actor.last_name DESC, actor.first_name DESC
```

WHERE CLAUSE

Belangrijke info

LOGISCHE OPERATOREN: and, or, IN, BETWEEN

- Selecteer alle velden uit de tabel actor waar de familienaam gelijk is aan 'ZELLWEGER'

```
SELECT *
```

```
FROM actor
```

```
WHERE actor.last_name = 'ZELLWEGER'
```

- Selecteer alle velden uit de tabel actor waar de familienaam gelijk is aan 'ZELLWEGER' of 'GUINNESS'

oplossing 1:

```
SELECT *
```

```
FROM actor
```

```
WHERE actor.last_name = 'ZELLWEGER' or actor.last_name = 'GUINNESS'
```

oplossing 2:

```
SELECT *
```

```
FROM actor
```

```
WHERE actor.last_name IN ('ZELLWEGER', 'GUINNESS')
```

CURSUS MYSQL

- Selecteer alle velden uit de tabel actor waar de familienaam begint met Z of met D

```
SELECT *
FROM actor
WHERE actor.last_name LIKE ('Z%') OR actor.last_name LIKE ('D%')
```

- Selecteer alle velden uit de tabel actor waar de familienaam DAVIS, DUKAKIS of ZELLWEGER is.

```
SELECT *
FROM actor
WHERE actor.last_name IN ('DAVIS', 'DUKAKIS', 'ZELLWEGER')
```

- Selecteer alle acteurs waar het id groter of gelijk is dan 5 en kleiner of gelijk is aan 10.

oplossing 1:

```
SELECT *
FROM actor
WHERE actor.actor_id >= 5 and actor.actor_id <= 10
```

oplossing 2:

```
SELECT *
FROM actor
WHERE actor.actor_id BETWEEN 5 and 10
```

FUNCTIONS Belangrijke info

AS: het keyword AS wordt gebruikt om een veld een anderen naam te geven in een weergave of een nieuw samengesteld veld een naam te geven. AS is dus wat noemt een **alias** van een bestaand of nieuw veld.

In MySQL kunnen we daarnaast ook functies toepassen. Functies worden toegepast op bestaande velden in. Functies worden gekenmerkt door de naamgeving en ronde haakjes. Bijvoorbeeld: de functie MIN() haalt de laagste waarde uit een kolom.

Hieronder zie je een lijst van veel gebruikte functies

COUNT	De MySQL COUNT-aggregatiefunctie wordt gebruikt om het aantal rijen in een databasetabel te tellen.
MAX	Met de MySQL MAX-aggregatiefunctie kunnen we de hoogste (maximale) waarde voor een bepaalde kolom selecteren.
MIN	Met de MySQL MIN-aggregatiefunctie kunnen we de laagste (minimum) waarde voor een bepaalde kolom selecteren.
AVG	De MySQL AVG-aggregatiefunctie selecteert de gemiddelde waarde voor een bepaalde tabelkolom.
SUM	Met de MySQL SUM-aggregatiefunctie kunt u het totaal voor een numerieke kolom selecteren.
SQRT	Dit wordt gebruikt om een vierkantswortel van een bepaald getal te genereren.
RAND	Dit wordt gebruikt om een willekeurig getal te genereren met behulp van de MySQL-opdracht.
CONCAT	Dit wordt gebruikt om elke tekenreeks binnen een MySQL-opdracht samen te voegen .
LENGTH	Geeft de lengte van een string terug
CURDATE	Retourneert de huidige datum
CURTIME	Retourneert de huidige tijd
ADDDATE	Telt dagen bij een datum op
ADDTIME	Telt tijd bij tijd op
DATEFORMAT	Hier kan je de weergave van een datum mee wijzigen
DATE_SUB	Hier kan je 2 data van elkaar aftrekken
DATE	Geeft de datum van vandaag terug
WEEKDAY	Retourneert het weekdag nummer (index)
WEEK	Retourneert het weeknummer (index)

MySQL telt dus heel wat BUILT-IN functions. Er zijn er een paar honderd. Alle functies kun je hieronder terugvinden via de volgende link:

[MySQL :: MySQL 8.4 Reference Manual :: 14.1 Built-In Function and Operator Reference](#)

- Selecteer het hoogste bedrag uit de tabel payment
`SELECT max(amount)
 from payment`
- Hoeveel rijen telt de tabel payment
`SELECT max(amount)
 from payment`

- Selecteer de gemiddelde prijs van de bedragen uit de tabel payment

```
SELECT avg(amount)
FROM payment
```

KЛАSSИКАLE OEFENINGEN

1. Film Titels in Hoofdletters

Schrijf een query die de titels van alle films in hoofdletters retourneert.

Query:

```
SELECT UPPER(title) AS film_titel
FROM film;
```

2. Lengte van de Film Titel

Haal de titels van alle films op en geef ook de lengte van elke titel weer.

Query:

```
SELECT title, LENGTH(title) AS titel_lengte
FROM film;
```

3. Film Titels zonder Spaties aan de Randen

Geef de film titels weer waarbij eventuele leidende en afsluitende spaties worden verwijderd.

Query:

```
SELECT TRIM(title) AS titel_bijgesneden
FROM film;
```

4. Films die een Bepaalde Tekst bevatten

Haal de titels van alle films op die het woord 'love' in hun titel hebben (ongeacht hoofdletters).

Query:

```
SELECT title
FROM film
WHERE LOWER(title) LIKE '%love%';
```

5. Eerste en Laatste Teken van Acteursnamen

Toon de voornaam van elke acteur, samen met het eerste en laatste teken van hun voornaam.

Query:

```
SELECT first_name, LEFT(first_name, 1) AS eerste_karakter, RIGHT(first_name, 1) AS laatste_karakter
FROM actor;
```

```
SELECT count(distinct last_name) as aantal
```

```
FROM actor
```

GROUP BY

 Belangrijke info

Een group by groepeert rijen/velden die identiek zijn aan elkaar tot één rij.

Let op: bij een GROUP BY dien je altijd ALLE VELDEN over te nemen die in het SELECT gedeelte staan ZONDER de velden die van een functie zouden voorzien zijn.

Voorbeeld van een group by :

ADAM JONES

ADAM JONES

BRIDGET SLOAN

DENNIS HARPER



ADAM JONES

BRIDGET SLOAN

DENNIS HARPER

- Selecteer alle familienamen uit de tabel actor en voeg daarna een tweede kolom toe die het aantal per naam weergeeft. Sorteer dan volgens het aantal van Z-A.

```
SELECT last_name, count(last_name) as aantal
FROM actor
GROUP BY last_name
ORDER BY aantal DESC
```

- Selecteer alle familienamen en voornamen uit de tabel actor en voeg daarna een tweede kolom toe die het aantal per naam weergeeft. Sorteer dan volgens het aantal van Z-A.

```
SELECT last_name, first_name, count(last_name) as aantal
FROM actor
GROUP BY last_name, first_name
ORDER BY aantal DESC
```

HAVING

 Belangrijke info

Een having clause wordt gebruikt wanneer we uit het resultaat van een group by dienen te FILTEREN. Een having is eigenlijk de where voor een GROUP BY

- Selecteer alle familienamen uit de actor tabel. Geef dan de lijst weer met familienamen die meer dan 1 x voorkomen!

CURSUS MYSQL

```
select last_name  
from actor  
group by last_name  
having count(*) > 1
```

Oefening 1

Vraag: Selecteer alle verschillende rental_rate waarden uit de film tabel.

```
SELECT DISTINCT rental_rate  
FROM film;
```

Oefening 2

Vraag: Toon alle actors met de achternaam SMITH of JONES.

```
SELECT *  
FROM actor  
WHERE last_name = 'SMITH' OR last_name = 'JONES';
```

Oefening 3

Vraag: Selecteer de film_id en title van alle films met een rental_rate van meer dan 2.99, gesorteerd op title.

```
SELECT film_id, title  
FROM film  
WHERE rental_rate > 2.99  
ORDER BY title;
```

Oefening 4

Vraag: Tel het aantal films per rating en toon enkel diegenen met meer dan 100 films.

```
SELECT rating, COUNT(*) AS film_count  
FROM film  
GROUP BY rating  
HAVING COUNT(*) > 100;
```

Oefening 5

Vraag: Selecteer alle customer_id waar de email NULL is.

```
SELECT customer_id  
FROM customer  
WHERE email IS NULL;
```

Oefening 6

Vraag: Selecteer alle films (title) met een rental_duration van 5 dagen en een replacement_cost tussen 15 en 25.

```
SELECT title  
FROM film  
WHERE rental_duration = 5 AND replacement_cost BETWEEN 15 AND 25;
```

Oefening 7

Vraag: Toon alle inventory_id's waarvan de store_id niet gelijk is aan 2.

```
SELECT inventory_id  
FROM inventory  
WHERE NOT store_id = 2;
```

Oefening 8

Vraag: Selecteer de verschillende store_id waarden uit de staff tabel.

```
SELECT DISTINCT store_id  
FROM staff;
```

Oefening 9

Vraag: Selecteer het aantal klanten (customer_id) per store_id.

```
SELECT store_id, COUNT(customer_id) AS customer_count  
FROM customer  
GROUP BY store_id;
```

Oefening 10

Vraag: Toon alle film titels waarvan de replacement_cost groter is dan 20, gesorteerd op aflopende replacement_cost.

```
SELECT title  
FROM film  
WHERE replacement_cost > 20  
ORDER BY replacement_cost DESC;
```

Oefening 11

Vraag: Selecteer alle category_id's van de category tabel waarvan de name niet Children is.

```
SELECT category_id  
FROM category  
WHERE NOT name = 'Children';
```

Oefening 12

Vraag: Tel het aantal inventory_id per film_id en toon enkel resultaten met meer dan 20 exemplaren.

```
SELECT film_id, COUNT(inventory_id) AS inventory_count  
FROM inventory  
GROUP BY film_id  
HAVING COUNT(inventory_id) > 20;
```

Oefening 13

Vraag: Selecteer de actor_id en first_name voor acteurs met de achternaam die begint met K.