Kennis Check blok 6

Kennis Check blok 6.1

1. Wat is een database en waarom gebruiken we die?

Een database is een digitale 'schatkist' waarin je gegevens over bijvoorbeeld studenten, producten of klanten netjes opslaat.

Je gebruikt een database omdat het overzichtelijk is, snel werkt en voorkomt dat je dingen dubbel bijhoudt.

2. Hoe zet je ‘echte wereld’ gegevens om naar tabellen en kolommen?

Je kijkt naar de onderwerpen (zoals 'Student', 'Opleiding') en noemt die entiteiten. Van elk onderwerp maak je kolommen (attributen), zoals naam, klas of geboortedatum. Zo maak je structuur in je data.

3. Wat is het verschil tussen ruwe gegevens en een gestructureerd model?

Ruwe gegevens zijn zoals je ze tegenkomt: alle info door elkaar. Een gestructureerd model brengt orde: je verdeelt info over tabellen met duidelijke kolommen. Daardoor raak je minder snel in de war.

4. Waarom is het foutgevoelig om gegevens meerdere keren op te slaan?

Stel dat je opleiding ‘Software Developer’ vijf keer typt en één keer tikfout maakt. Dan heb je verschillende versies van dezelfde opleiding – dat maakt fouten en verwarring. Daarom bewaar je info op één plek.

5. Wat is een entiteit en wat is een attribuut?

Een entiteit is een 'dingen'-soort, zoals Student of Cursus. Attributen zijn eigenschappen daarvan, zoals naam, klas of duur. Zo kun je entiteiten en eigenschappen herkennen en verwerken.

6. Waarom hebben we altijd een primary key nodig?

Een primary key is een uniek kenmerk in een tabel, bijvoorbeeld student_id. Zo weet je zeker dat elke rij uniek is en kun je records makkelijk terugvinden en koppelen.

7. Wat is een 1:N-relatie en hoe maak je die in een ERD?

Bij een 1:N-relatie hoort één entiteit bij meerdere bij een andere. Bijvoorbeeld één klant -> meerdere boekingen. Je tekent een haakje aan de kant met “veel” en zet een foreign key in de ‘veel’-tabel.

8. Wat is een foreign key en waarom gebruiken we die?

Een foreign key is een verwijzing in de ene tabel naar de primary key van een andere tabel. Daarmee maak je een verband tussen twee tabellen, zoals boeken gekoppeld aan klanten.

9. Wat zijn datatypes en waarom moet je die zorgvuldig kiezen?

Datatypes bepalen wat voor soort gegevens je opslaat, zoals INT, VARCHAR of DATE. Als je het verkeerde datatype kiest, kan je database fout gaan of traag worden. En voor telefoon gebruik je VARCHAR omdat het niet rekent.

10. Wat is een N:N-relatie en hoe los je dat op in je model?

Een N:N-relatie is een relatie waarin meerdere records van entiteit A gekoppeld zijn aan meerdere van B, bijvoorbeeld leerlingen met meerdere lessen. Dat los je op met een tussentabel waarin de foreign keys van beide entiteiten samen komen.

SQL

1. Waarom gebruik je verschillende SQL-commando's zoals SELECT, INSERT, UPDATE en DELETE?

Omdat je met elke opdracht iets anders doet met de gegevens. Met `SELECT` haal je gegevens op, met `INSERT` voeg je iets toe, met `UPDATE` verander je iets, en met `DELETE` verwijder je iets.

Als je maar één commando zou hebben, zou je alles zelf moeten bouwen. Nu is het veel makkelijker en duidelijker.

2. Wat gebeurt er als je een WHERE-clausule vergeet bij een UPDATE- of DELETE-opdracht?

Dan pas je alles aan of verwijder je alles in de hele tabel! Dat is meestal niet wat je wilt. De `WHERE` zorgt ervoor dat je alleen die rijen verandert die aan een bepaalde voorwaarde voldoen, bijvoorbeeld: alleen de student met ID 5.

3. Waarom gebruik je JOIN?

Stel: je hebt een tabel met studenten en een andere met klasnamen. Als je wilt zien in welke klas een student zit, moet je de tabellen combineren. Dat doe je met een `JOIN`. Dan kun je gegevens uit beide tabellen tegelijk laten zien, alsof het één lijst is.

4. Stel: je hebt een tabel 'studenten' en je wilt alleen de studenten uit Amsterdam zien, welke SQL-structuur gebruik je?

Dan gebruik je `WHERE woonplaats = 'Amsterdam'` achter je SELECT.

Zo zeg je: "Geef alleen de rijen waarvan de woonplaats gelijk is aan Amsterdam."

5. Wat is het nut van een wildcard zoals '%' in een LIKE-zoekopdracht?

De `%` betekent: 'maakt niet uit wat hier staat'. Als je zoekt op `naam LIKE 'J%'`, krijg je alle namen die beginnen met een J, zoals "Jesse", "Julia" of "Joris". Het is handig als je niet precies weet wat iemand heeft ingevuld.

6. Waarom is het belangrijk om kolommen te kiezen in SELECT, in plaats van SELECT *?

Met `SELECT *` haal je alles op, ook dingen die je niet nodig hebt. Dat kan je website trager maken en is onduidelijk. Als je alleen de kolommen kiest die je nodig hebt, is je code netter en sneller.

7. Wat verandert er aan je data-structuur door een PRIMARY KEY toe te voegen?

Een `PRIMARY KEY` zorgt ervoor dat elke rij uniek is, bijvoorbeeld op ID.

De rij die je een `PRIMARY KEY` geeft, moet dus ook uniek zijn; van elke waarde mag er maar één voorkomen.

8. Hoe zou jij een fout in een query ontdekken en oplossen?

Lees goed wat de foutmelding zegt. Vaak is het een typefout, een komma die mist of een fout in de naam van een kolom. Probeer de query stap voor stap te testen. Gebruik bijvoorbeeld

phpMyAdmin om te kijken of je query daar werkt.

Als je er niet uit komt dan kan je de query ook stapje voor stapje eenvoudiger maken, door telkens iets weg te laten.

9. Wat is het verschil tussen een SQL-query schrijven in phpMyAdmin en dezelfde query uitvoeren via PHP/PDO?

In phpMyAdmin voer je de query handmatig in – één keer. Met PHP/PDO doe je dat in je code, automatisch, telkens als iemand je website gebruikt. Zo kun je gegevens tonen of opslaan zonder handmatig iets te doen.

PDO

Waarom kiezen we voor PDO boven oudere methodes zoals mysql_* of mysqli?

PDO is veiliger en flexibeler. Het helpt je om makkelijker over te stappen naar een andere database (bijvoorbeeld van MySQL naar SQLite).

PDO is het meest recent en modern. PDO wordt verder ontwikkeld en mysqli `minder.

PDO is iets eenvoudiger in gebruikt.

Wat is een DSN in PDO?

DSN betekent "Data Source Name". Dat is een stukje tekst waarin staat met welke database je verbinding maakt.

Waarom is het handig om de verbinding (bijvoorbeeld connection.php) in een apart bestand te zetten?

Zo hoef je niet overal dezelfde code te typen. Als er iets verandert (zoals het wachtwoord), dan hoef je dat maar op één plek aan te passen. Je maakt je code netter, overzichtelijker en makkelijker te onderhouden.

Wat is het verschil tussen de database-instellingen op je eigen laptop en op een echte server?

Op je laptop gebruik je vaak "localhost" en een gebruiker (*root*) zonder wachtwoord.

Op een echte server is dat niet veilig, dus daar gebruik je andere gegevens. Het is belangrijk dat je die gegevens netjes gescheiden houdt, anders werkt je site straks niet online of wordt hij gehackt.

Wat is het verschil tussen query() en prepare() + execute()?

Met `query()` stuur je direct een SQL-opdracht.

Met `prepare()` + `execute()` gebruik je een veilige manier waarbij je eerst zegt "wat je ongeveer wilt doen" en daarna de gegevens pas invult. Dat is veiliger, vooral bij gegevens van een formulier.

Hoe helpt prepare() om SQL-injecties te voorkomen?

Een SQL-injectie is als iemand iets raars intypt in een formulier om je database kapot te maken. `prepare()` zorgt ervoor dat die invoer niet als SQL-code wordt gezien, maar gewoon als tekst. Daardoor kunnen hackers niks kapot maken.

Hoe haal je meerdere rijen uit een database met PDO?

Je doet eerst `$stmt = $conn->query("SELECT ...")`, dan gebruik je `fetch()` om één rij te krijgen of `fetchAll()` voor alles tegelijk.

`fetch()` is handig als je maar één resultaat verwacht, zoals bij een login. `fetchAll()` gebruik je als je bijvoorbeeld een lijst van producten wilt laten zien.

?? Opdracht

Maak de kennis-check.

? Inleveren

Aan het einde van de kennis-check ontvang je een certificaat. Maak een schermafdruk en lever deze in.

7. Wat is een 1:N-relatie en hoe maak je die in een ERD?

Bij een 1:N-relatie hoort één entiteit bij meerdere bij een andere. Bijvoorbeeld één klant -> meerdere boekingen. Je tekent een haakje aan de kant met “veel” en zet een foreign key in de ‘veel’-tabel.

8. Wat is een foreign key en waarom gebruiken we die?

Een foreign key is een verwijzing in de ene tabel naar de primary key van een andere tabel. Daarmee maak je een verband tussen twee tabellen, zoals boeken gekoppeld aan klanten.

9. Wat zijn datatypes en waarom moet je die zorgvuldig kiezen?

Datatypes bepalen wat voor soort gegevens je opslaat, zoals INT, VARCHAR of DATE. Als je het verkeerde datatype kiest, kan je database fout gaan of traag worden. En voor telefoon gebruik je VARCHAR omdat het niet rekent.

10. Wat is een N:N-relatie en hoe los je dat op in je model?

Een N:N-relatie is een relatie waarin meerdere records van entiteit A gekoppeld zijn aan meerdere van B, bijvoorbeeld leerlingen met meerdere lessen. Dat los je op met een tussentabel waarin de foreign keys van beide entiteiten samen komen.

SQL

1. Waarom gebruik je verschillende SQL-commando's zoals SELECT, INSERT, UPDATE en DELETE?

Omdat je met elke opdracht iets anders doet met de gegevens. Met `SELECT` haal je gegevens op, met `INSERT` voeg je iets toe, met `UPDATE` verander je iets, en met `DELETE` verwijder je iets.

Als je maar één commando zou hebben, zou je alles zelf moeten bouwen. Nu is het veel makkelijker en duidelijker.

2. Wat gebeurt er als je een WHERE-clausule vergeet bij een UPDATE- of DELETE-opdracht?

Dan pas je alles aan of verwijder je alles in de hele tabel! Dat is meestal niet wat je wilt. De `WHERE` zorgt ervoor dat je alleen die rijen verandert die aan een bepaalde voorwaarde voldoen, bijvoorbeeld: alleen de student met ID 5.

3. Waarom gebruik je JOIN?

Stel: je hebt een tabel met studenten en een andere met klasnamen. Als je wilt zien in welke klas een student zit, moet je de tabellen combineren. Dat doe je met een `JOIN`. Dan kun je gegevens uit beide tabellen tegelijk laten zien, alsof het één lijst is.

4. Stel: je hebt een tabel 'studenten' en je wilt alleen de studenten uit Amsterdam zien, welke SQL-structuur gebruik je?

Dan gebruik je `WHERE woonplaats = 'Amsterdam'` achter je SELECT.

Zo zeg je: "Geef alleen de rijen waarvan de woonplaats gelijk is aan Amsterdam."

5. Wat is het nut van een wildcard zoals '%' in een LIKE-zoekopdracht?

De `%` betekent: 'maakt niet uit wat hier staat'. Als je zoekt op `naam LIKE 'J%'`, krijg je alle namen die beginnen met een J, zoals "Jesse", "Julia" of "Joris". Het is handig als je niet precies weet wat iemand heeft ingevuld.

6. Waarom is het belangrijk om kolommen te kiezen in SELECT, in plaats van SELECT *?

Met `SELECT *` haal je alles op, ook dingen die je niet nodig hebt. Dat kan je website trager maken en is onduidelijk. Als je alleen de kolommen kiest die je nodig hebt, is je code netter en sneller.

7. Wat verandert er aan je data-structuur door een PRIMARY KEY toe te voegen?

Een `PRIMARY KEY` zorgt ervoor dat elke rij uniek is, bijvoorbeeld op ID.

De rij die je een `PRIMARY KEY` geeft, moet dus ook uniek zijn; van elke waarde mag er maar één voorkomen.

8. Hoe zou jij een fout in een query ontdekken en oplossen?

Lees goed wat de foutmelding zegt. Vaak is het een typefout, een komma die mist of een fout in de naam van een kolom. Probeer de query stap voor stap te testen. Gebruik bijvoorbeeld

phpMyAdmin om te kijken of je query daar werkt.

Als je er niet uit komt dan kan je de query ook stapje voor stapje eenvoudiger maken, door telkens iets weg te laten.

9. Wat is het verschil tussen een SQL-query schrijven in phpMyAdmin en dezelfde query uitvoeren via PHP/PDO?

In phpMyAdmin voer je de query handmatig in – één keer. Met PHP/PDO doe je dat in je code, automatisch, telkens als iemand je website gebruikt. Zo kun je gegevens tonen of opslaan zonder handmatig iets te doen.

PDO

Waarom kiezen we voor PDO boven oudere methodes zoals mysql_* of mysqli?

PDO is veiliger en flexibeler. Het helpt je om makkelijker over te stappen naar een andere database (bijvoorbeeld van MySQL naar SQLite).

PDO is het meest recent en modern. PDO wordt verder ontwikkeld en mysqli `minder.

PDO is iets eenvoudiger in gebruikt.

Wat is een DSN in PDO?

DSN betekent "Data Source Name". Dat is een stukje tekst waarin staat met welke database je verbinding maakt.

Waarom is het handig om de verbinding (bijvoorbeeld connection.php) in een apart bestand te zetten?

Zo hoef je niet overal dezelfde code te typen. Als er iets verandert (zoals het wachtwoord), dan hoef je dat maar op één plek aan te passen. Je maakt je code netter, overzichtelijker en makkelijker te onderhouden.

Wat is het verschil tussen de database-instellingen op je eigen laptop en op een echte server?

Op je laptop gebruik je vaak "localhost" en een gebruiker (*root*) zonder wachtwoord.

Op een echte server is dat niet veilig, dus daar gebruik je andere gegevens. Het is belangrijk dat je die gegevens netjes gescheiden houdt, anders werkt je site straks niet online of wordt hij gehackt.

Wat is het verschil tussen query() en prepare() + execute()?

Met `query()` stuur je direct een SQL-opdracht.

Met `prepare()` + `execute()` gebruik je een veilige manier waarbij je eerst zegt "wat je ongeveer wilt doen" en daarna de gegevens pas invult. Dat is veiliger, vooral bij gegevens van een formulier.

Hoe helpt prepare() om SQL-injecties te voorkomen?

Een SQL-injectie is als iemand iets raars intypt in een formulier om je database kapot te maken. `prepare()` zorgt ervoor dat die invoer niet als SQL-code wordt gezien, maar gewoon als tekst. Daardoor kunnen hackers niks kapot maken.

Hoe haal je meerdere rijen uit een database met PDO?

Je doet eerst `$stmt = $conn->query("SELECT ...")`, dan gebruik je `fetch()` om één rij te krijgen of `fetchAll()` voor alles tegelijk.

`fetch()` is handig als je maar één resultaat verwacht, zoals bij een login. `fetchAll()` gebruik je als je bijvoorbeeld een lijst van producten wilt laten zien.

Kennis Check blok 6.2

?? Opdracht

Maak de kennis-check.

? Inleveren

Aan het einde van de kennis-check ontvang je een certificaat. Maak een schermafdruk en lever deze in.

--

Revision #12
Created 2025-06-20 08:15:27 UTC by Max
Updated 2026-07-10 09:10:00 UTC by Max