Hand-out

Programmeren

Periode 2

06. Query’s

1. **Inleiding**

   In dit hoofdstuk gaan we query's behandelen, we nemen dit onderwerp even apart omdat query's toepasbaar zijn op iedere database en in de basis per database ongeveer de zelfde structuur hebben. Zowel SQL als MySQL databases gebruiken nagenoeg de zelfde query's, met hier en daar een uitzondering.

2. **Wat is een Query**

   Een query is een regel waarmee je de database aanstuurt. Je kunt iets toevoegen, aanpassen of verwijderen, dit doe je door een query aan te maken en deze uit te voeren op je database. De gegevens die worden toegevoegd aangepast en of verwijderd heten records, in alle gevallen gebruik je een query dus op een of meer records die onderdeel uitmaken van een of meerdere tabellen maar altijd in een database, nooit meerdere.

   In de volgende stukken gaan we de meest voorkomende statements behandelen.

3. **Het CREATE statement**

   Via dit statement kun je een database creëren maar je kunt er ook een tabel mee creëren.

   Om een database te creëren gebruik je het volgende statement :

   ```
   CREATE DATABASE testDB;
   ```

   Om een tabel te creëren gebruik je :

   ```
   CREATE TABLE Persons (
       PersonID int,
       LastName varchar(255),
       FirstName varchar(255),
       Address varchar(255),
       City varchar(255)
   );
   ```

4. **Het SELECT statement**

   SELECT wordt gebruikt wanneer je een of meerdere records uit een of meerdere tabellen wil selecteren. Selecteren houdt in, je krijgt een verzameling terug van deze records, en die bevatten data. Deze data kun je daarna binnen je applicatie of website gebruiken. Het meest simpele select statement ziet er als volgt uit :

   ```
   SELECT * FROM table_name;
   ```

In het bovenstaande geval haal je met de asterix (*) alle velden (kolommen) van de tabel op. Je kunt ook specifieker zijn door de velden die je op wil halen te specificeren.

```
SELECT CustomerName, City FROM Customers;
```

Binnen het SELECT statement hebben we een aantal opties :

a. **TOP**

   Top wordt gebruikt wanneer je het maximaal aantal records wil aangeven berekend vanaf het eerste record.

   ```
   SELECT * FROM Customers TOP 10;
   ```

   Dit selecteert de eerste 10 records uit de tabel Customers.

b. **WHERE**

   Dit gebruiken we wanneer we een specifieke selectie willen maken, bijvoorbeeld alle records met het type 5.

   ```
   SELECT * FROM Customers WHERE type=5 TOP 10;
   ```

c. **AND OR NOT**

   Dit passen we binnen een query toe wanneer we exclusiviteit willen hebben, als we het een of het ander willen hebben, of het een en het ander.

   ```
   SELECT * FROM Customers WHERE type=5 OR type=6 AND brand=’some’;
   ```

d. **INNER JOIN**

   Met een INNER JOIN kun je velden uit meerdere tabellen in het zelfde resultaat krijgen, vandaar ook de naam JOIN. Naast de INNER JOIN hebben we nog LEFT, RIGHT, FULL en SELF JOIN, hier een aantal voorbeelden :

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

   Om drie tabellen te kopellen zou je het volgende kunnen doen :

   ```
   SELECT Orders.OrderID, Customers.CustomerName, Shippers.ShipperName
   FROM ((Orders
   INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID)
   INNER JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID);
   ```

5. **Het INSERT INTO statement**

   Met dit statement kun je records aan de database toevoegen. Hierbij moet je letten op twee dingen, dat je goed aangeeft welke velden je wil gaan vullen en dat je de data (VALUES) in de zelfde volgorde als de velden zet, als volgt :

```
INSERT INTO Customers
(CustomerName, ContactName, Address, City, PostalCode, Country)
VALUES
('Cardinal', 'Tom B.', 'Skagen 21', 'Stavanger', '4006', 'Norway');
```

6. **Het UPDATE statement**

   Je kunt ook records updaten, d.w.z. je kunt gegevens veranderen in je database. Als je dit doet dan is het niet zo dat je de gegevens daarna automatisch terug ziet, je zult opnieuw een SELECT statement uit moeten voeren.

   Bij het UPDATE statement kun je net als bij SELECT de extra opties gebruiken als WHERE, AND OR NOT, behalve TOP. Als volgt :

```
UPDATE Customers
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
WHERE CustomerID = 1;
```

7. **Het DELETE statement**

   Als je records wil verwijderen dan kan dat, dit doe je door middel van het DELETE statement. Let wel op met het deleten van gegevens, deze zijn daarna niet meer herstelbaar.

```
DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';
```

8. **Het DROP DATABASE statement**

   Met dit statement moet je enorm oppassen, hier kun je namelijk een database of een tabel mee verwijderen.

```
DROP DATABASE testDB;
```

9. **Demo**

   In dit stuk gaan we klassikaal een demo maken. Het is de bedoeling dat je mee kijkt en zo stapsgewijs houvast krijgt op het onderwerp, maak daar waar nodig is aantekeningen in je schrift of in je kladblok. Als je klaar bent ga je zelf aan de slag dus let goed op!

10. **Opdracht**

In deze opdracht ga je een database maken voor het restaurant “Ham & Eggs”. Het restaurant heeft gevraagd of er een nieuw menu kaart kan worden bedacht met nieuwe prijzen. Het menu is onderverdeeld in de volgende categorie :

- Voorgerechten
- Soepen
- Hoofdgerechten
- Nagerechten
- Desserts
- Dranken

De database moet gevuld worden met data, welke data mag je zelf bedenken, zorg in ieder geval dat je per groep minimaal 5 opties hebt. Dat je koppel tabellen gebruikt om relaties te maken en let op dat je de prijzen ten alle tijden aan moet kunnen passen (dus zet deze in een aparte tabel) en dat je het menu moet kunnen uitbreiden.

Als je klaar bent met het design van je database en het vullen van de data MOET je met **een** SELECT query alle menu items in een keer aan kunnen tonen met prijzen en categorie, pas als je dit kunt ben je klaar met de opdracht.