**Database commands**

**Tabel creeeren met NOT NULL CONSTRAINT**

CREATE TABLE Tabel02 (

Voornaam varchar(20) NOT NULL,

Achternaam varchar(20) NOT NULL );

INSERT INTO Tabel02 (Voornaam,Achternaam) VALUES ('Jan' , 'de Boer');

INSERT INTO Tabel02 (Voornaam) VALUES ('Jan');

INSERT INTO Tabel02 (Achternaam) VALUES ('Visser');

SELECT \* FROM Tabel02;

**Tabel creeeren met UNIQUE KEY CONSTRAINT**

CREATE TABLE Tabel04 (

Voornaam varchar(20),

Achternaam varchar(20)

CONSTRAINT UK\_Tabel04 UNIQUE (Voornaam, Achternaam) );

INSERT INTO Tabel04 (Voornaam,Achternaam) VALUES ('Jan','de Boer');

INSERT INTO Tabel04 (Voornaam,Achternaam) VALUES ('Jan','de Boer');

SELECT \* FROM Tabel04;

**TABEL VERANDEREN MET PK CONSTRAINT**

use sander;

create table student1(

studentnummer int NOT NULL UNIQUE,

Voornaam varchar(20),

Achternaam varchar(20),

Alter table student add constraint UK\_student unique (Voornaam, Achternaam);

**KOLOM, TABEL EN DATABASE VERWIJDEREN**

delete from student where Voornaam='Klaas';

drop table student1;

drop table student;

use GamingGame;

drop database sander;

**AGREGRATIEFUNCTIES**

use Northwind;

select \*

from Employees

select count(\*) from Employees;

select \* from Products;

select sum(UnitsOnOrder) from Products;

select \* from Products

select AVG(UnitsOnOrder) from Products;

select \* from Products

select max(UnitsOnOrder) from Products;

select \* from Products

select MIN(UnitsOnOrder) from Products;

select \* from Products

select UnitPrice from Products;

select distinct(UnitPrice) from Products;

**PREDICATEN EN OPERATOREN**

select \* from Products;

select \* from Products where UnitPrice > 200 and UnitsInStock > 1;

select \* from Products;

select \* from Products where UnitsInStock <= 1

select \* from Products;

select \* from Products where SupplierID = 10;

select \* from Products where not SupplierID = 10;

select \* from Products;

select \* from Products where SupplierID in (1,3,5,7,9,11,13,15,17,19);

select \* from Products;

select \* from Products where UnitPrice between 30 and 40;

select \* from Suppliers;

select \* from Suppliers where Region is NULL

select \* from Suppliers;

select \* from Suppliers where ContactName like '%HAN%';

**check constraint**

CREATE TABLE TIENERS (Naam VARCHAR(20) PRIMARY KEY, Leeftijd INT

CONSTRAINT CHECK\_LEEFTIJD CHECK (Leeftijd > 12 AND Leeftijd < 20));

Een check constraint aan een tabel toevoegen.

ALTER TABLE TIENERS ADD CONSTRAINT CHECK\_NAAM CHECK (Naam != 'Peter');

ALTER TABLE Student ADD CONSTRAINT C\_Student CHECK ((StudentID IS NOT NULL)

AND Achternaam <>'Steen' AND Voornaam <>'Jan'

AND Geboortejaar BETWEEN 1900 AND 2000

AND Geboorteplaats != Woonplaats);

**Les 13 Subquerrys**

SELECT \* FROM Products WHERE UnitsInStock = (SELECT MAX(UnitsInStock) FROM Products);

SELECT SupplierID, ProductID, ProductName, ReorderLevel, UnitsInStock FROM Products

WHERE ProductID IN (SELECT ProductID FROM Products WHERE UnitsInStock <= ReorderLevel)

ORDER BY SupplierID;

SELECT FirstName, LastName, HireDate FROM Employees

WHERE HireDate = (SELECT MIN(HireDate) FROM Employees);

**Subquerrys waarbij je info uit 2 of meer tabellen haalt. (dit noem je geneste subquerrys)**

SELECT CustomerID, CompanyName, ContactName, ContactTitle, Address, City, Region,

PostalCode FROM Customers WHERE CustomerID IN

(SELECT DISTINCT CustomerID FROM Orders WHERE EmployeeID =

(SELECT EmployeeID FROM Employees WHERE LastName = 'Suyama'))

SELECT EmployeeID, FirstName, LastName FROM Employees

WHERE EmployeeID = (SELECT EmployeeID FROM Orders WHERE OrderID = (Select MIN(OrderID) from Orders));

**Les 14 View (CREATE, SELECT, DROP)**

CREATE VIEW vSalesRepresentatives AS

(SELECT EmployeeID, Title, LastName, FirstName, City FROM Employees WHERE Title = 'Sales Representative';)

SELECT FirstName, LastName, City FROM vSalesRepresentatives WHERE City = 'London';

DROP VIEW vSalesRepresentatives

CREATE VIEW vReorderProducts AS

(SELECT ProductID, ProductName FROM Products WHERE UnitsInStock <= ReorderLevel AND UnitsOnOrder = 0 AND Discontinued = 0)

SELECT \* FROM vReorderProducts;

**Les 15 inner en outer joins**

**impliciete joins**

SELECT Naam, Leeftijd, Tabel11.Woonplaats

FROM Tabel10, Tabel11

WHERE Tabel10.Woonplaats = Tabel11.Woonplaats;

**Expliciete joins ( de inner join maakt dit een expliciete join.)**

SELECT Naam, Leeftijd, Tabel11.Woonplaats

FROM Tabel10 INNER JOIN Tabel11

ON Tabel10.Woonplaats = Tabel11.Woonplaats;

**(dit heironder gaat verder met joins.)**

USE Northwind;

SELECT Employees.EmployeeID, CustomerID, OrderID

FROM Employees, Orders

WHERE Employees.EmployeeID = Orders.EmployeeID;

SELECT \* FROM Employees, Orders;

USE Northwind;

SELECT OrderID, CustomerID, Orders.EmployeeID, FirstName, LastName

FROM Orders, Employees

WHERE City = 'Seattle' or city = 'London' and orders.EmployeeID= Employees.EmployeeID

ORDER BY LastName, FirstName;

USE Northwind;

SELECT OrderID, Customers.CustomerID, CompanyName, ContactName

FROM Orders INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID

WHERE OrderID is NULL OR Customers.CustomerID is NULL

ORDER BY CompanyName;

**Les 15.3 pseudoniemen (hierbij kan je de AS weglaten.)**

SELECT Naam, Leeftijd, T11.Woonplaats

FROM Tabel11 AS T11, Tabel10 AS T10

WHERE T10.Woonplaats = T11.Woonplaats;

**Gebruik van pseudoniem waarbij je join gebruikt op 2x dezelfde tabel.**

SELECT \*

FROM Tabel11 T1, Tabel11 T2

WHERE T1.Woonplaats = 'Den Haag'

AND T1.Naam = 'Jan'

AND T2.Leeftijd <= T1.Leeftijd;

**Les 15.4 INNER JOIN (geeft alle rijen weer waarvan waarde uit beide kolommen gelijk zijn)**

**(hierbij kan je het woord inner weglaten)**

SELECT Naam, Leeftijd, Tabel11.Woonplaats

FROM Tabel10 INNER JOIN Tabel11

ON Tabel10.Woonplaats = Tabel11.Woonplaats;

**Les 15.5 outer join (kan waarde tonen in rij waarvan waarde wel in 1 tabel voorkomt maar niet in tabel 2)**

**de LEFT OUTER geeft aan dat rijen uit tabel links van JOIN gezien moeten worden waarvan waarde niet voorkomt in kolom rechts van JOIN**

SELECT Naam, Leeftijd, Tabel10.Woonplaats

FROM Tabel10 LEFT OUTER JOIN Tabel11

ON Tabel10.Woonplaats = Tabel11.Woonplaats;

**Les 15.6 RIGHT OUTER JOIN (zelfde als LEFT OUTER JOIN maar dan omgekeerd. Dus JOIN refereert het rechter kolom)**

**Hierbij kan je OUTER weglaten**

SELECT Naam, Leeftijd, Tabel10.Woonplaats

FROM Tabel11 RIGHT OUTER JOIN Tabel10

ON Tabel10.Woonplaats = Tabel11.Woonplaats;

**Les 15.7 FULL OUTER JOIN (Laat waarde zien van tabel zowel links als rechts van woord JOIN)**

SELECT Naam, Leeftijd, Tabel10.Woonplaats

FROM Tabel11b FULL OUTER JOIN Tabel10

ON Tabel10.Woonplaats = Tabel11b.Woonplaats;

**LES 15.8 CROSS JOIN (neemt aantal rijen van 1 tabel en vermenigvuldigt deze met rijen uit tabel 2 dus mix alle mogelijke waarde met elkaar en zet deze in tabel)**

SELECT \*

FROM Tabel10 CROSS JOIN Tabel11;

Post

## Database beheer opdrachten

**Les 23 CREATE LOGIN (creëert login voor gebruiker. gebruiker kan hiermee in database komen. dit moet je wel combineren met create user)**

CREATE LOGIN <loginnaam> WITH PASSWORD = '<paswoord>';

**CREATE USER** (**creëert user en login voor gebruiker. Gebruiker kan hiermee database bereiken)**

CREATE USER <usernaam> FOR LOGIN <loginnaam> WITH DEFAULT\_SCHEMA <schemanaam>; GO;

**CREATE SCHEMA ( creëert nieuwe schema in)**

CREATE SCHEMA <schema naam> AUTHORIZATION <username>;

**CREATE ROLE (creëert nieuwe database rol.)**

CREATE ROLE role\_name AUTHORIZATION [dbo];

ALTER ROLE role\_name ADD MEMBER user\_name;

**GRANT SELECT (toekennen van privileges aan tabel)**

GRANT SELECT|INSERT|UPDATE|DELETE ON

table\_name TO user\_name|role\_name;

**REVOKE SELECT (verwijderen van privileges aan tabel)**

REVOKE SELECT|INSERT|UPDATE|DELETE ON

table\_name FROM user\_name|role\_name;