

DATABANKEN - SQL

C# programmeur

SQL – INHOUD

- **Joins (vervolg)**
 - Inner – left join – right join
- **Union**
- **Subqueries**
- **Wijzigen van data**
 - **INSERT - UPDATE - DELETE**

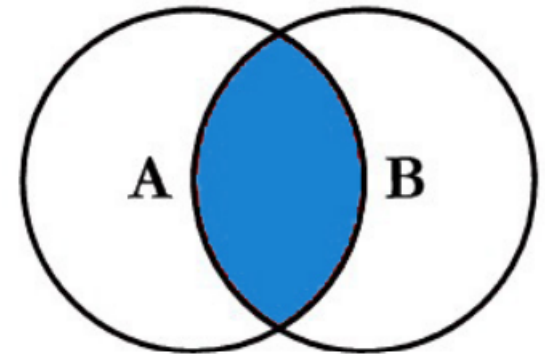
DATABANKEN - SQL

Selecteren uit meerdere tabellen - JOINS

Inner Join

Syntaxis

```
SELECT columns  
FROM table1  
INNER JOIN table2  
ON table1.column = table2.column;
```



TAKEN

NR	TAAK
1	AFWAS
3	TUIN
3	POST



PERSONEN

NUMMER	NAAM	PLAATS
1	JAN	GENT
2	PIET	BRUSSEL
3	JORIS	AALST
4	JAN	BRUSSEL

```
USE joinvoorbeeld;  
SELECT *  
FROM taken AS t  
INNER JOIN personen AS p  
ON t.nr = p.nummer;
```

Inner Join

TAKEN

NR	TAAK
1	AFWAS
3	TUIN
3	POST

1



PERSONEN

NUMMER	NAAM	PLAATS
1	JAN	GENT
2	PIET	BRUSSEL
3	JORIS	AALST
4	JAN	BRUSSEL

2

USE joinvoorbeeld;

SELECT *

FROM taken AS t

INNER JOIN personen AS p

ON t.nr = p.nummer;

1

2

NR	TAAK	NUMMER	NAAM	PLAATS
1	AFWAS	1	JAN	GENT
3	TUIN	3	JORIS	AALST
3	POST	3	JORIS	AALST

INNER Join

The screenshot shows a SQL IDE window titled "SQL File 5* x". The main editor contains two SQL queries. The first query uses a `where` clause to join `bieren` and `brouwers` tables. The second query uses the `INNER JOIN` and `ON` syntax for the same purpose. Below the editor, the "Result Set Filter" bar is empty. The results pane displays a table with two columns: "Naam" and "bmaam". The table contains 15 rows of data, including beer names and their respective brewers. The status bar at the bottom indicates "Result 18 x" and "Read Only !".

```
--Select uit meerdere tabellen
select naam, brnaam from bieren, brouwers
where bieren.BrouwerNr = brouwers.BrouwerNr

--andere notatie geeft zelfde resultaat:
select naam, brnaam from bieren INNER JOIN brouwers
ON bieren.BrouwerNr = brouwers.BrouwerNr
```

Naam	bmaam
A.C.O.	Steedje
Aalbeeks St. Corneliusbier (=Kapittel pater (Het))	Van Eecke
Aardbeien witbier	Huyghe
Aarschots kruikenbier (=St. Sebastiaan grand cru)	Sterkens
Abt Bijbier (Nen)	Domus
Adler	Haacht
Aerts 1900	Palm
Affligem blond (Abdij)	Smedt (De)
Affligem christmas ale (Abdij)	Smedt (De)
Affligem dubbel (Abdij)	Smedt (De)
Affligem patersvat	Smedt (De)
Affligem tripel (Abdij)	Smedt (De)

SELF JOIN = INNER Join van tabel met zichzelf

- Records combineren uit dezelfde tabel
- Verplicht aliassen voor de tabel gebruiken

Voorbeeld

Een tabel met personeelsleden. Ieder personeelslid heeft een backup onder de vorm van een nummer die verwijst naar een ander personeelslid

PERSONEELSLEDEN

NUMMER	NAAM	PLAATS	BACKUP
1	JAN	GENT	2
2	PIET	BRUSSEL	1
3	JORIS	AALST	4
4	JAN	BRUSSEL	3
5	MARINA	LEUVEN	2

Joris uit Aalst is de backup van Jan uit Brussel.

Piet uit Brussel is zowel de backup van Jan uit Gent als van Marina uit Leuven

...

We willen nu een lijst van alle personeelsleden met hun backups. We zijn daarbij niet geïnteresseerd in de nummers. We willen dus de namen en de plaatsen zien.

SELF JOIN = INNER Join van tabel met zichzelf

PERSONEELSLEDEN alias PERS

NUMMER	NAAM	PLAATS	BACKUP
1	JAN	GENT	2
2	PIET	BRUSSEL	1
3	JORIS	AALST	4
4	JAN	BRUSSEL	3
5	MARINA	LEUVEN	2



PERSONEELSLEDEN alias BACK

NUMMER	NAAM	PLAATS	BACKUP
1	JAN	GENT	2
2	PIET	BRUSSEL	1
3	JORIS	AALST	4
4	JAN	BRUSSEL	3
5	MARINA	LEUVEN	2

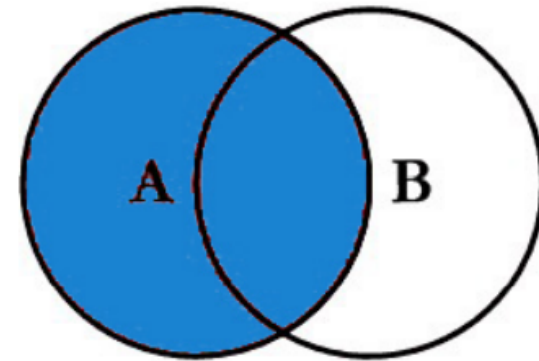


WERKNEMER	BACKUP
JAN UIT GENT	PIET UIT BRUSSEL
PIET UIT BRUSSEL	JAN UIT GENT
JORIS UIT AALST	JAN UIT BRUSSEL
JAN UIT BRUSSEL	JORIS UIT AALST
MARINA UIT LEUVEN	PIET UIT BRUSSEL

Left Join

Syntaxis

```
SELECT columns  
FROM table1  
LEFT [OUTER] JOIN table2  
ON table1.column = table2.column;
```



TAKEN

NR	TAAK
1	AFWAS
3	TUIN
3	POST



PERSONEN

NUMMER	NAAM	PLAATS
1	JAN	GENT
2	PIET	BRUSSEL
3	JORIS	AALST
4	JAN	BRUSSEL

```
USE joinvoorbeeld;  
SELECT *  
FROM taken AS t  
LEFT OUTER JOIN personen AS p  
ON t.nr = p.nummer;
```

Left Join

PERSONEN

NUMMER	NAAM	PLAATS
1	JAN	GENT
2	PIET	BRUSSEL
3	JORIS	AALST
4	JAN	BRUSSEL



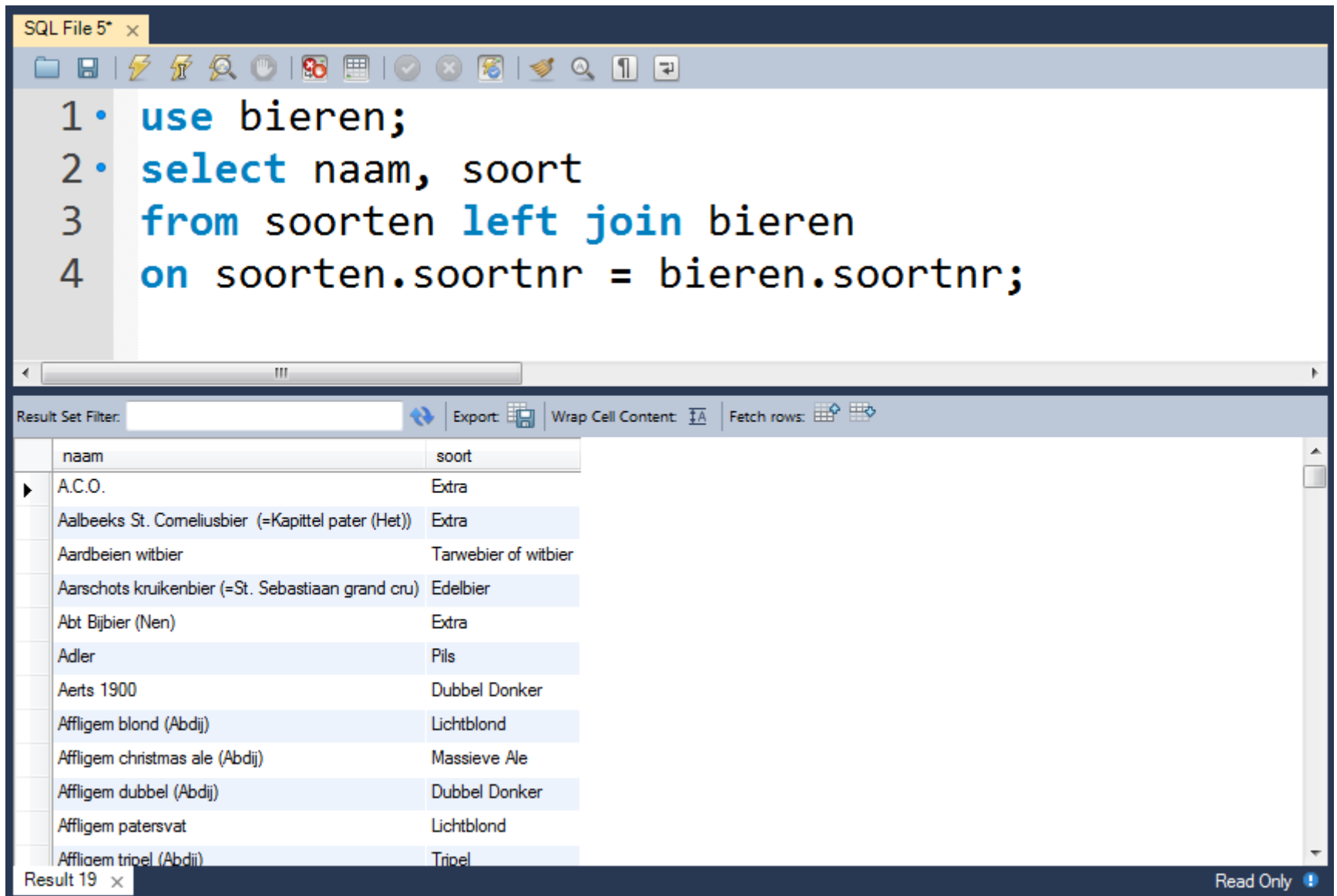
TAKEN

NR	TAAK
1	AFWAS
3	TUIN
3	POST

SELECT * **FROM** taken AS t
LEFT OUTER JOIN personen AS p **ON** t.nr = p.nummer;

NUMMER	NAAM	PLAATS	NR	TAAK
1	JAN	GENT	1	AFWAS
2	PIET	BRUSSEL	NULL	NULL
3	JORIS	AALST	3	TUIN
3	JORIS	AALST	3	POST
4	JAN	BRUSSEL	NULL	NULL

JOIN : left join



The screenshot shows a SQL IDE window titled "SQL File 5* x". The editor contains the following SQL query:

```
1 • use bieren;  
2 • select naam, soort  
3   from soorten left join bieren  
4   on soorten.soortnr = bieren.soortnr;
```

Below the editor, the "Result Set Filter:" section is empty. The "Export:" button is visible. The "Wrap Cell Content:" button is also visible. The "Fetch rows:" section shows a grid icon and a refresh icon.

The results are displayed in a table with two columns: "naam" and "soort". The table contains 19 rows of data, starting with "A.C.O." and "Extra".

naam	soort
A.C.O.	Extra
Aalbeeks St. Corneliusbier (=Kapittel pater (Het))	Extra
Aardbeien witbier	Tarwebier of witbier
Aarschots kruikenbier (=St. Sebastiaan grand cru)	Edelbier
Abt Bijbier (Nen)	Extra
Adler	Pils
Aerts 1900	Dubbel Donker
Affligem blond (Abdij)	Lichtblond
Affligem christmas ale (Abdij)	Massieve Ale
Affligem dubbel (Abdij)	Dubbel Donker
Affligem patersvat	Lichtblond
Affligem tripel (Abdij)	Tripel

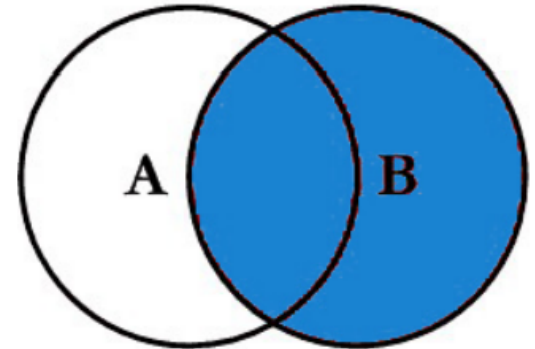
Result 19 x

Read Only

JOIN : right join

Syntaxis

```
SELECT columns  
FROM table1  
RIGHT [OUTER] JOIN table2  
ON table1.column = table2.column;
```



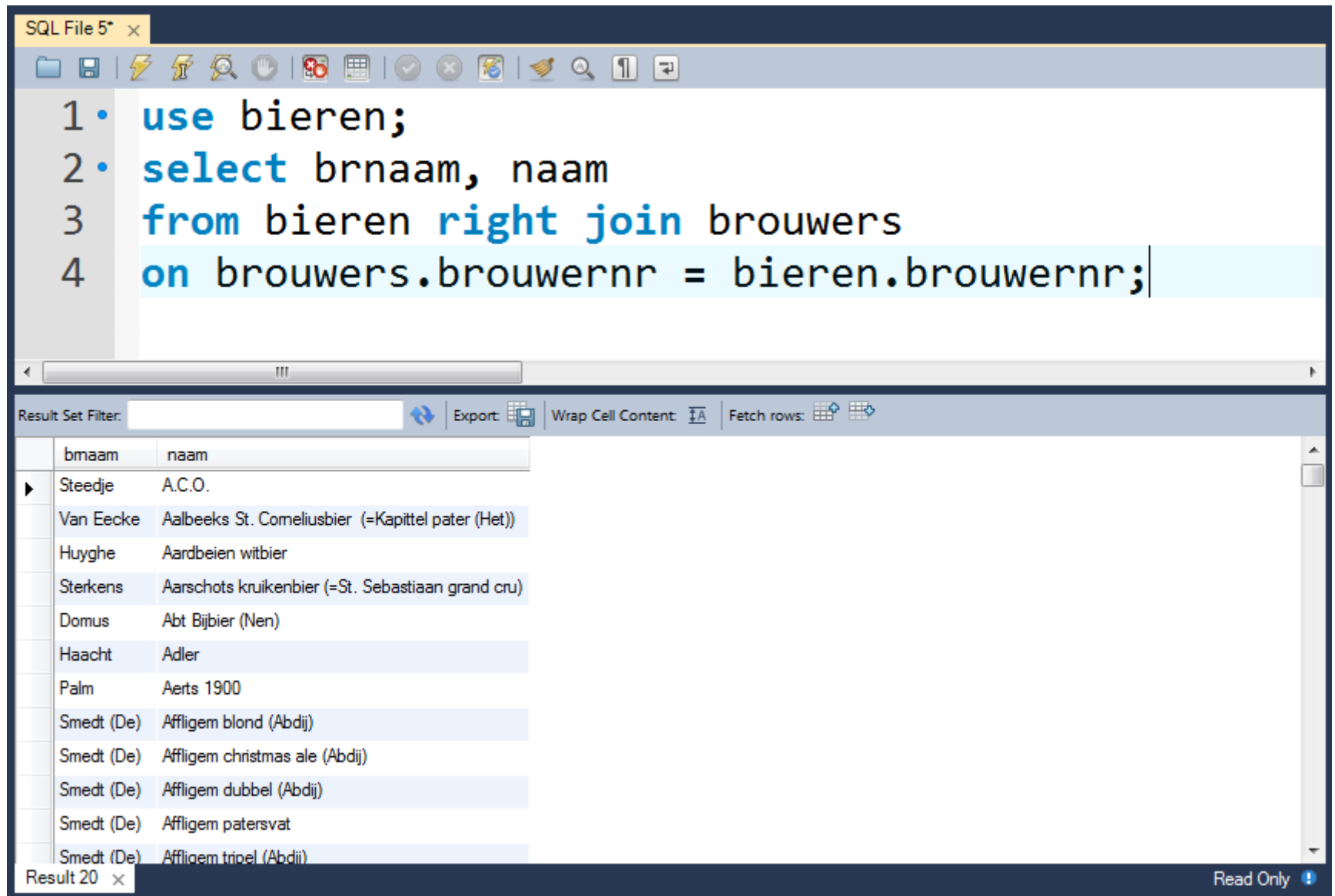
Werkt net hetzelfde als de LEFT JOIN maar dan in omgekeerde volgorde

```
SELECT columns  
FROM table1  
RIGHT [OUTER] JOIN table2  
ON table1.column = table2.column;
```



```
SELECT columns  
FROM table2  
LEFT [OUTER] JOIN table1  
ON table1.column = table2.column;
```

JOIN : right join



The screenshot shows a SQL IDE window titled "SQL File 5* x". The editor contains the following SQL query:

```
1 • use bieren;  
2 • select brnaam, naam  
3 from bieren right join brouwers  
4 on brouwers.brouwernr = bieren.brouwernr;
```

Below the editor is a toolbar with icons for file operations, search, and execution. The "Result Set Filter" is empty. The "Export" button is active. The "Wrap Cell Content" button is active. The "Fetch rows" button is active. The results pane shows a table with two columns: "brnaam" and "naam". The table contains 20 rows of data, with the first row expanded to show details.

brnaam	naam
Steedje	A.C.O.
Van Eecke	Aalbeeks St. Comeliusbier (=Kapittel pater (Het))
Huyghe	Aardbeien witbier
Sterkens	Aarschots kruikenbier (=St. Sebastiaan grand cru)
Domus	Abt Bijbier (Nen)
Haacht	Adler
Palm	Aerts 1900
Smedt (De)	Affligem blond (Abdij)
Smedt (De)	Affligem christmas ale (Abdij)
Smedt (De)	Affligem dubbel (Abdij)
Smedt (De)	Affligem patersvat
Smedt (De)	Affligem tripel (Abdij)

The bottom of the window shows a "Result 20 x" tab and a "Read Only" status indicator.

UNION

SQL File 5*

```
1 • use bieren;  
2 • select * from bieren where soortnr = 12  
3 union  
4 select * from bieren where soortnr = 5;
```

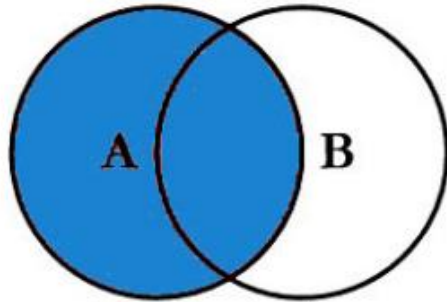
Result Set Filter: [] Edit: [] Export/Import: [] Wrap Cell Content: []

	BierNr	Naam	BrouwerNr	SoortNr	Alcohol
▶	155	Blok-bok (Nen)	33	12	7.00
	292	Chouffe-bok 6666	1	12	6.66
	484	Eupener klosterbier special bock	41	12	6.50
	1188	Sezoens quattro	70	5	7.00
	1475	Veteren alt	30	5	8.00
*	NULL	NULL	NULL	NULL	NULL

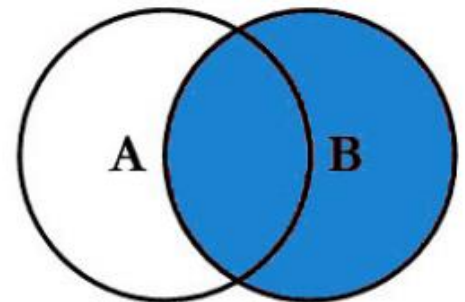
bieren 33 x

Apply Cancel

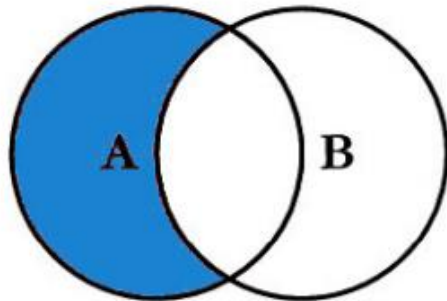
SQL JOINS



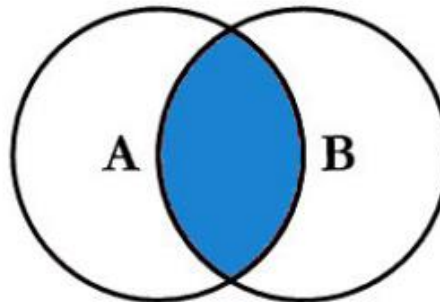
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



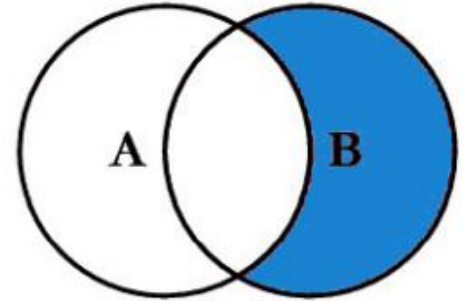
```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



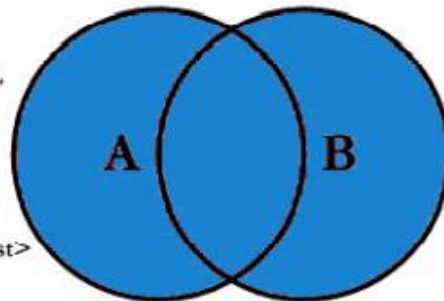
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```



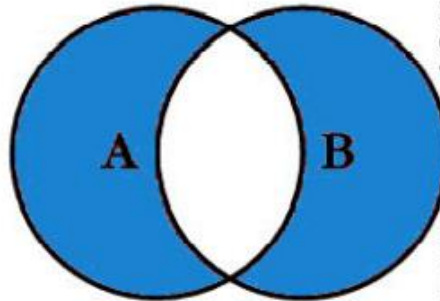
```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

DATABANKEN - SQL

Oefeningen Selectie – Deel 4

Oefeningen JOINS

Download van Github: [LESSEN_DATABANK\LES 5](#)

1. Oefeningen-INNER-LEFT-RIGHT JOINS.sql

DATABANKEN - SQL

Selecteren uit meerdere tabellen - Subqueries

Subqueries

OUTER QUERY



```
SELECT lastname, firstname  
FROM employees  
WHERE officeCode IN ( SELECT officeCode  
                        FROM offices  
                        WHERE country = 'BELGIUM' )
```



SUBQUERY of INNER QUERY

Subqueries die één waarde teruggeven

Als een subquery een **enkelvoudige waarde** terug geeft, kan je deze gebruiken in een **WHERE** clause met een **vergelijkingsoperator**:

Voorbeeld

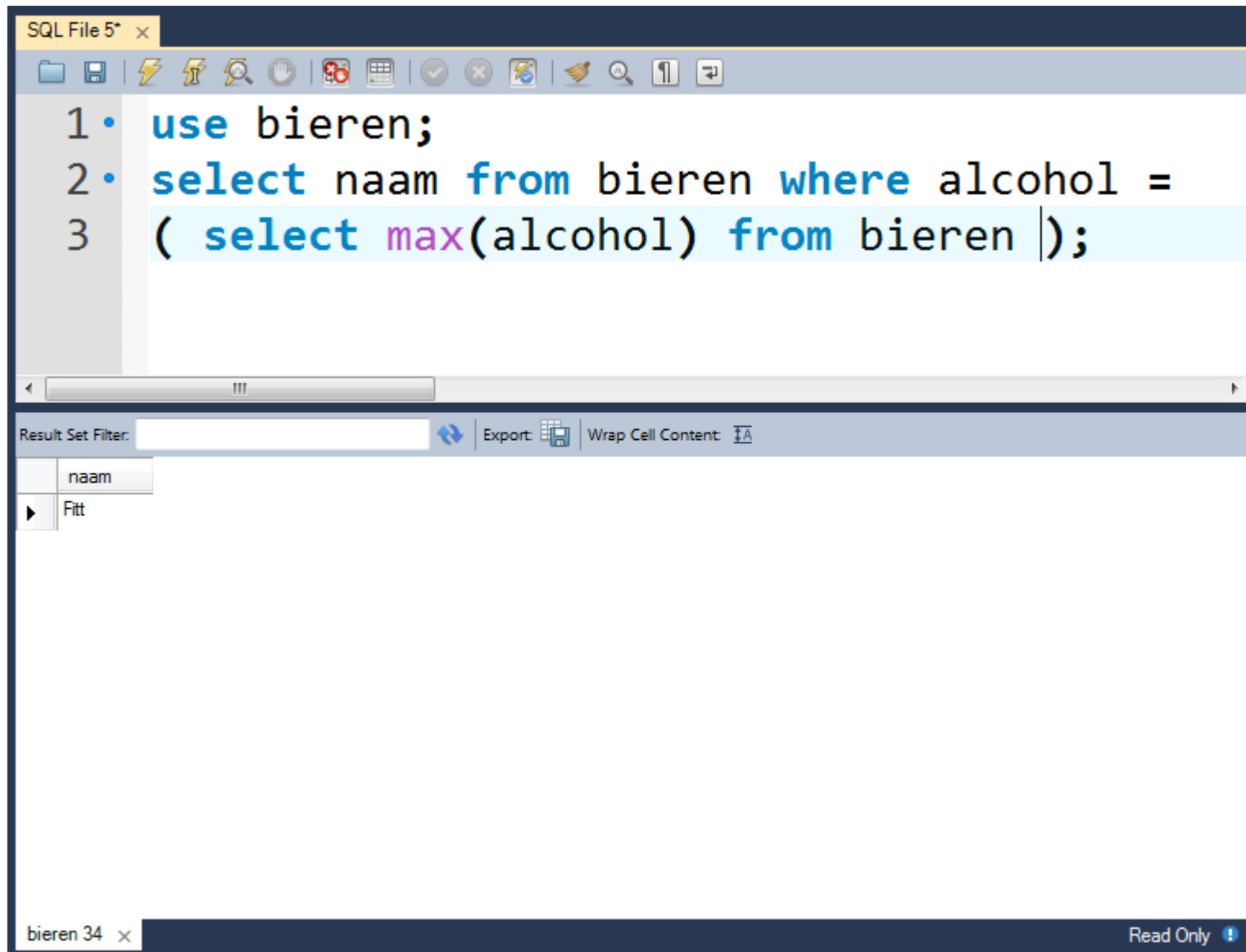
Database: classicmodels

Geef de klant die het meeste heeft betaald.

```
SELECT customerNumber,  
        checkNumber,  
        amount  
FROM payments  
WHERE amount = (  
    SELECT MAX(amount)  
    FROM payments  
)
```

	customerNumber	checkNumber	amount
►	141	JE105477	120166.58
*	NULL	NULL	NULL

Subquery die één waarde teruggeeft - voorbeeld



The screenshot shows a SQL IDE window titled "SQL File 5* x". The query editor contains the following SQL code:

```
1 • use bieren;  
2 • select naam from bieren where alcohol =  
3   ( select max(alcohol) from bieren );
```

Below the query editor is a toolbar with icons for file operations, execution, and formatting. The "Result Set Filter" section is empty. The "Export" button is visible. The "Wrap Cell Content" button is also visible. The result set is displayed in a table with two columns: "naam" and "Fitt". The table is currently empty.

At the bottom of the window, the status bar shows "bieren 34 x" and "Read Only !".

Subqueries die meerdere waarden teruggeven

Als een subquery een **meerder waarden** terug geeft, kan je deze gebruiken in een **WHERE** clause met andere operatoren zoals **IN** of **NOT IN**:

Voorbeeld

Geef een lijst op van alle klanten die tot nu toe nog geen enkel produkt hebben besteld.

	klantNr	checkNr	bedrag
►	112	HQ55022	32641.98
	112	ND748579	33347.88
	114	GG31455	45864.03
	114	MA765515	82261.22

Query

```
SELECT customername  
FROM customers  
WHERE customerNumber NOT IN (  
  SELECT DISTINCT customernumbe  
  FROM orders  
)
```

Subquery die meerdere waarden teruggeeft -vb

```
USE [BierenDb];  
  
Select * from bieren  
where SoortNr NOT IN  
(select SoortNr from soorten where Naam = 'Alcoholvrij');
```

132 %

Results Messages

	BierNr	Naam	BrouwerNr	SoortNr	Alcohol
1	4	A.C.O.	104	18	7
2	5	Aalbeeks St. Corneliusbier (=Kapittel pater (Het))	113	18	6.5
3	7	Aardbeien witbier	56	53	2.5
4	8	Aarschots kruikenbier (=St. Sebastiaan grand cru)	105	15	7.6
5	10	Abt Bijbier (Nen)	33	18	7
6	11	Adler	51	42	6.75
7	12	Aerts 1900	81	14	7
8	13	Affligem blond (Abdij)	100	33	7
9	14	Affligem christmas ale (Abdij)	100	36	9
10	15	Affligem dubbel (Abdij)	100	14	7
11	16	Affligem patersvat	100	33	7
12	17	Affligem tripel (Abdij)	100	59	8.5
13	18	Akila pilsener	68	42	5
14	20	Aldegonde brune	72	36	8.5
15	21	Aldegonde cuvee	72	15	7.5

Oefening subqueries

Opdracht 1

Geef een lijst van alle bieren met het hoogste alcoholpercentage.

Oplossing ?

Opdracht 2

Geef een lijst van alle bieren die in Brugge gebrouwen zijn.

Oplossing ?

Gecorreleerde subqueries

- Een correlated subquery is m.a.w. afhankelijk van de outer query.
- Een correlated subquery wordt éénmaal geëvalueerd voor elke rij van de outer query

Voorbeeld

```
SELECT productname, buyprice
FROM products AS p1
WHERE buyprice > (
    SELECT AVG(buyprice)
    FROM products
    WHERE productline = p1.productline)
```

	productname	buyprice
▶	1952 Alpine Renault 1300	98.58
	1996 Moto Guzzi 1100i	68.99
	2003 Harley-Davidson Eagle Drag Bike	91.02
	1972 Alfa Romeo GTA	85.68
	1962 LanciaA Delta 16V	103.42
	1968 Ford Mustang	95.34

Gecorreleerde subqueries

Voorbeeld

```
SELECT customerNumber klantenNr, customerName klant
FROM    customers
WHERE    customerNumber IN(
    SELECT customerNumber
    FROM orders
    WHERE orderNumber IN (
        SELECT orderNumber
        FROM orderdetails
        WHERE ( priceEach * quantityOrdered > 10000)
        GROUP BY orderNumber
    )
);
```

Een correlated subquery gebruikt informatie van de outer query

Gecorreleerde subqueries - voorbeeld bierendb

The screenshot shows a SQL IDE window titled "SQL File 5* x". The SQL editor contains the following query:

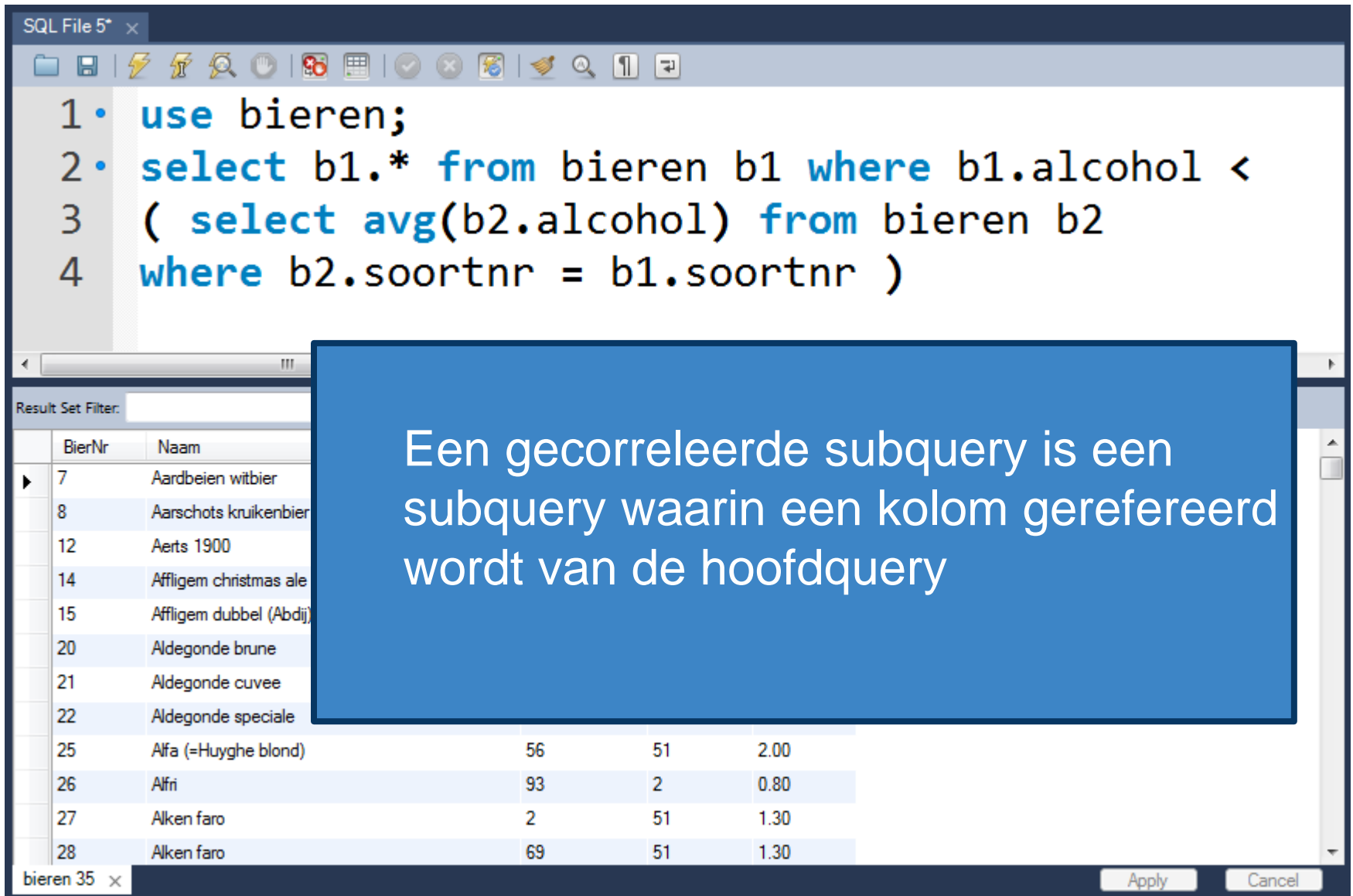
```
1 • use bieren;  
2 • select b1.* from bieren b1 where b1.alcohol <  
3   ( select avg(b2.alcohol) from bieren b2  
4   where b2.soortnr = b1.soortnr )
```

Below the editor, the "Result Set Filter:" is empty. The results are displayed in a table with the following columns: BierNr, Naam, BrouwerNr, SoortNr, and Alcohol.

BierNr	Naam	BrouwerNr	SoortNr	Alcohol
7	Aardbeien witbier	56	53	2.50
8	Aarschots kruikenbier (=St. Sebastiaan grand cru)	105	15	7.60
12	Aerts 1900	81	14	7.00
14	Affligem christmas ale (Abdij)	100	36	9.00
15	Affligem dubbel (Abdij)	100	14	7.00
20	Aldegonde brune	72	36	8.50
21	Aldegonde cuvee	72	15	7.50
22	Aldegonde speciale	72	36	8.50
25	Alfa (=Huyghe blond)	56	51	2.00
26	Alfri	93	2	0.80
27	Alken faro	2	51	1.30
28	Alken faro	69	51	1.30

At the bottom left, a tab labeled "bieren 35 x" is visible. At the bottom right, there are "Apply" and "Cancel" buttons.

Gecorreleerde subqueries



The screenshot shows a SQL IDE window titled "SQL File 5* x". The main text area contains the following SQL code:

```
1 • use bieren;  
2 • select b1.* from bieren b1 where b1.alcohol <  
3   ( select avg(b2.alcohol) from bieren b2  
4   where b2.soortnr = b1.soortnr )
```

Below the code, there is a "Result Set Filter:" section and a table of results. The table has columns "BierNr" and "Naam". The results are as follows:

BierNr	Naam
7	Aardbeien witbier
8	Aarschots kruikenbier
12	Aerts 1900
14	Affligem christmas ale
15	Affligem dubbel (Abdij)
20	Aldegonde brune
21	Aldegonde cuvee
22	Aldegonde speciale
25	Alfa (=Huyghe blond)
26	Alfri
27	Alken faro
28	Alken faro

At the bottom left, there is a tab labeled "bieren 35 x". At the bottom right, there are "Apply" and "Cancel" buttons.

Een gecorreleerde subquery is een subquery waarin een kolom gerefereerd wordt van de hoofdquery

Joins en SubQueries - Oefeningen

Download van Github:

Nieuwe database Plantv

Plantv_log.ldf

Plantv.mdf

Attach de database Plantv in SSMS

Beschrijvig van de tabellen: PlantvDb-beschrijving.pdf

En maak de volgende oefeningen:

Oefening-Plantv Database MS SQL Svr-Diagram.docx

Oefeningen JOIN deel4 en Subquery deel5.pdf

VRAGEN?