

# H8 Werken met meerdere tabellen.

Group by  
Join  
Union

**HO  
GENT**

# Basisvorm SELECT-statement

- SELECT voor raadplegen van 1 tabel

```
SELECT [ALL | DISTINCT] {*/uitdrukking [,uitdrukking ...]}  
FROM tabelnaam  
[WHERE voorwaarde(n)]  
[GROUP BY kolomnaam [,kolomnaam ...]  
[HAVING voorwaarde(n)]  
[ORDER BY {kolomnaam/volgnr}{ASC/DESC}[,...]]
```

- SELECT clause: specificeert de kolommen die je wenst te zien. DISTINCT zorgt ervoor dat de getoonde rijen alle uniek zijn
- FROM clause: geeft aan uit welke tabel de gegevens afkomstig zijn
- WHERE clause: opgave van de voorwaarden waaraan de getoonde rijen moeten voldoen
- ORDER BY clause: bepaalt de volgorde waarin de rijen getoond moeten worden
- GROUP BY en HAVING clause: groeperen van de gegevens



Group by en statistische functies.

# Statistische functies

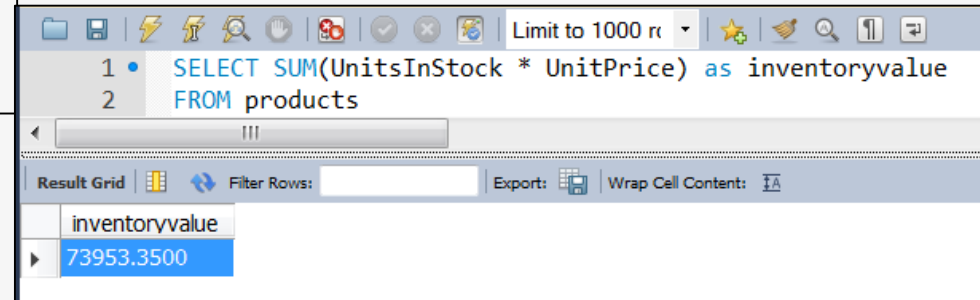
- Statistische functies (aka aggregate functions)
  - SQL voorziet 5 standaardfuncties
    - **SUM**(uitdrukking): som
    - **AVG**(uitdrukking): gemiddelde
    - **MIN**(uitdrukking): minimum
    - **MAX**(uitdrukking): maximum
    - **COUNT**(\*[**DISTINCT**] kolomnaam): aantal
  - Deze functies geven 1 antwoord per kolom (of groep) en mogen dus niet in een WHERE clause gebruikt worden.

# Som en gemiddelde

- SUM

- Retourneert het totaal van NIET NULL numerieke waarden in één kolom
- Enkel te gebruiken met numerieke argumenten
- Voorbeeld: Geef de totale stockwaarde.

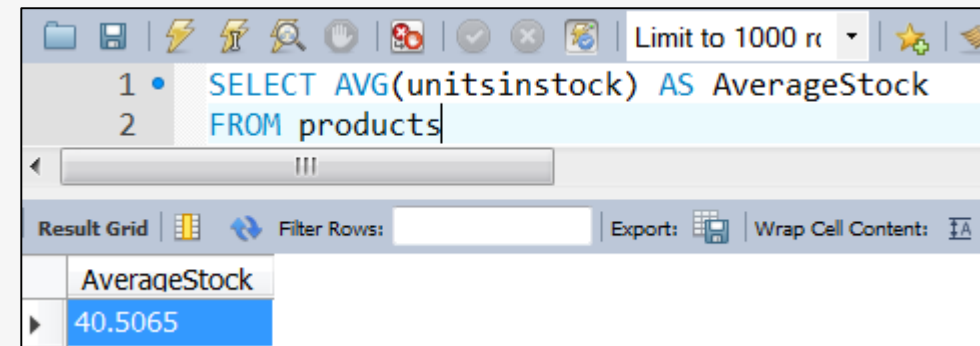
```
SELECT SUM(UnitsInStock * UnitPrice) as inventoryvalue  
FROM products
```



- AVG

- Retourneert het gemiddelde van NIET NULL numerieke waarden in een kolom
- Enkel te gebruiken met numerieke argumenten
- *Voorbeeld: Wat is het gemiddeld aantal producten in stock?*

```
SELECT AVG(unitsinstock) AS AverageStock  
FROM products
```

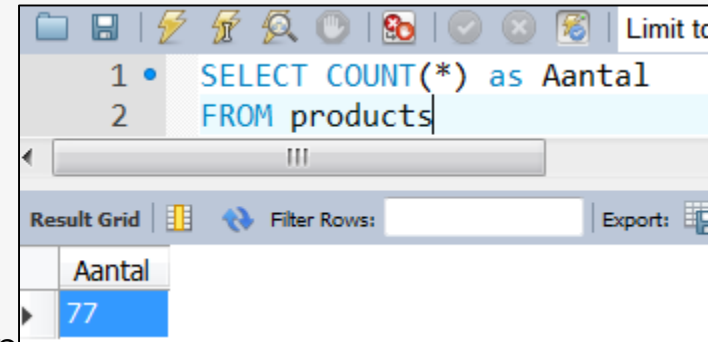


# Aantal rijen tellen

- COUNT

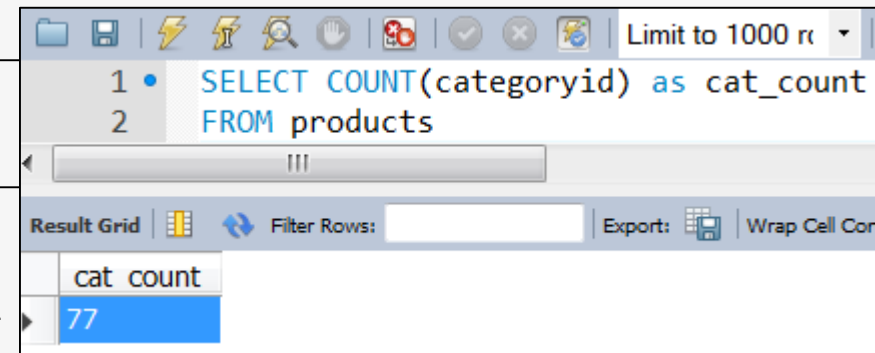
- Retourneert het aantal rijen, of een aantal waarden in een kolom
  - **COUNT(\*)** - telt het aantal rijen van de selectie
  - Voorbeeld: tel het aantal producten (dit is het aantal rijen).

```
SELECT COUNT(*) as Aantal  
FROM products
```



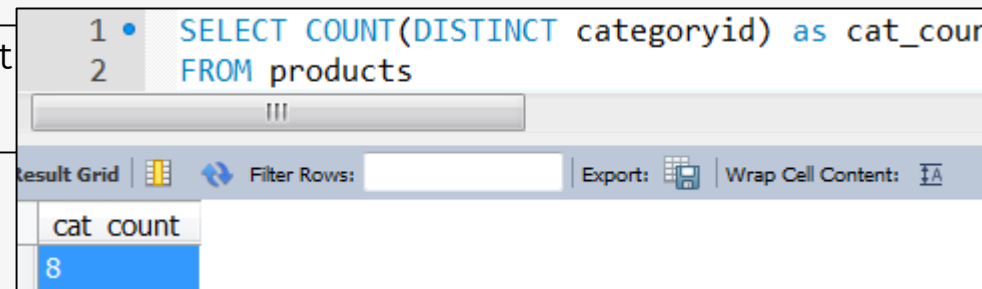
- **COUNT(kolomnaam)** - telt het aantal niet-lege velden in een kolom
- Voorbeeld: Tel het aantal NIET NULL waarden in de kolom categoryid.

```
SELECT COUNT(categoryid) as cat_count  
FROM products
```



- **COUNT(DISTINCT kolomnaam)** - telt het aantal verschillende niet-lege velden in een kolom
- Voorbeeld: Tel het aantal verschillende NIET NULL categorieën (categoryid) in products.

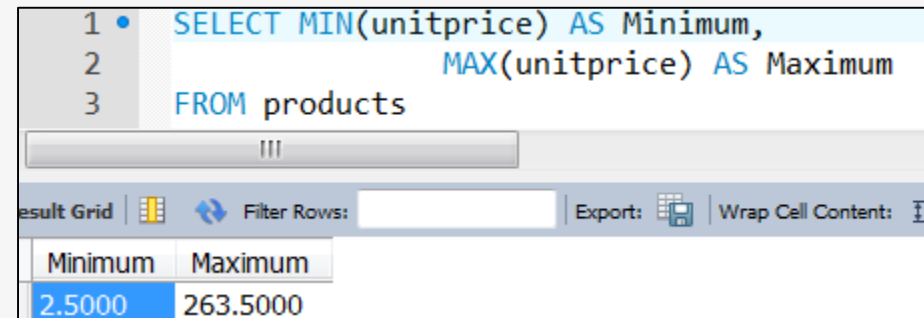
```
SELECT COUNT(DISTINCT categoryid) as cat_count  
FROM products
```



# Minimum en maximum

- MIN en MAX
  - retourneert de kleinste en de grootste waarde in een kolom
  - gelden zowel op numerieke als alfanumerieke argumenten
  - Voorbeeld: wat is het goedkoopste en het duurste product (= wat is de kleinste en de grootste eenheidsprijs)?

```
SELECT MIN(unitprice) AS Minimum,  
       MAX(unitprice) AS Maximum  
FROM products
```



The screenshot shows a SQL query editor with the following code:

```
1 • SELECT MIN(unitprice) AS Minimum,  
2       MAX(unitprice) AS Maximum  
3 FROM products
```

Below the editor is a 'Result Grid' with the following data:

Minimum	Maximum
2.5000	263.5000

The interface also includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Content' checkbox.

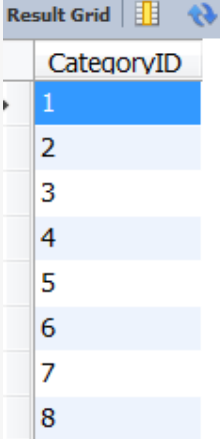
## Opmerkingen

1. Omdat een statistische functie maar **1 antwoord** oplevert, moeten ofwel alle uitdrukkingen in de SELECT clausule een statistische functie bevatten, ofwel geen enkele! Dit verandert wanneer we group-by introduceren...
2. Statistische functies houden geen rekening met **NULL waarden**.  
Uitzondering : COUNT(\*) (telt ook rijen die null waarden bevatten)

# Groeperen via GROUP BY

- Groeperen - Statistische functies over meerdere groepen.
  - **GROUP BY** clause :
    - Indeling van tabel in **groepen van rijen** met gemeenschappelijke kenmerken.
    - Per groep ontstaat 1 unieke rij (voor de resultset)!
    - Voorbeeld: Tot welke categorieën behoren de producten?

```
SELECT CategoryID  
FROM Products  
GROUP BY CategoryID
```



CategoryID
1
2
3
4
5
6
7
8

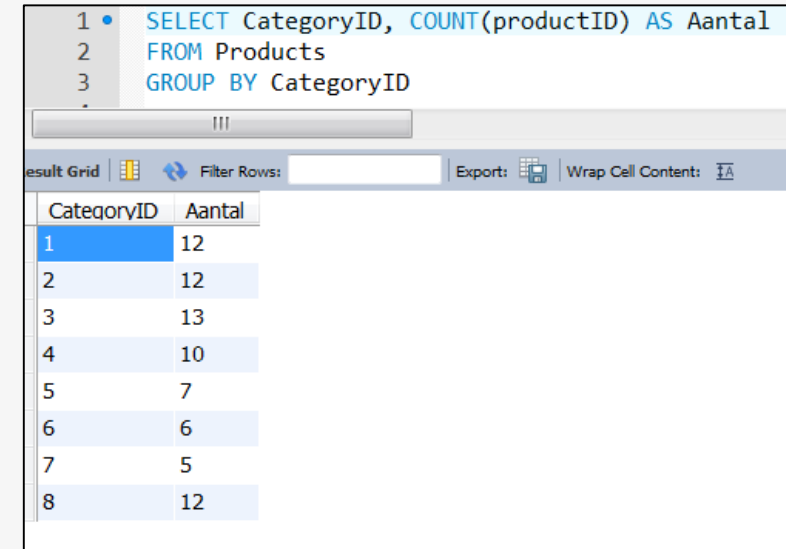
- Elke groep is een afzonderlijke verzameling waarop eventueel statistische bewerkingen (functies) uitgevoerd kunnen worden.
- De kolomnamen vermeld in de GROUP BY mogen nu ook samen met de statistische functies in de SELECT voorkomen.



# Groeperen via GROUP BY

- Enkele voorbeelden
  - Toon per categorie het aantal producten.

```
SELECT CategoryID, COUNT(productID) AS Aantal
FROM Products
GROUP BY CategoryID
```



The screenshot shows a SQL query editor with the following query:

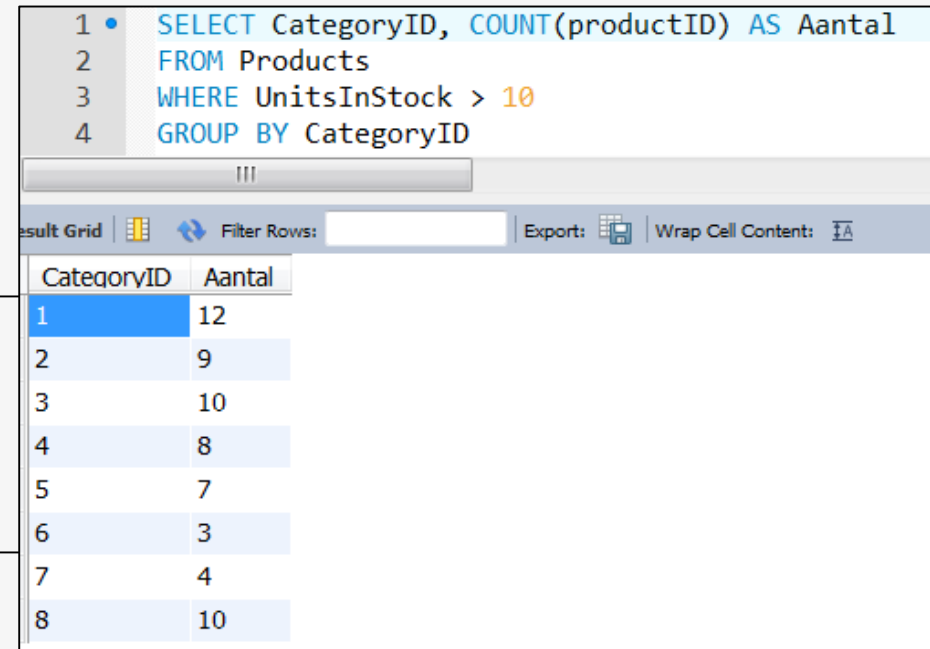
```
1 • SELECT CategoryID, COUNT(productID) AS Aantal
2 FROM Products
3 GROUP BY CategoryID
```

Below the query editor, there is a toolbar with buttons for 'Filter Rows', 'Export', and 'Wrap Cell Content'. Below the toolbar is a table with the following data:

CategoryID	Aantal
1	12
2	12
3	13
4	10
5	7
6	6
7	5
8	12

- Toon per categorie het aantal producten, waarvan er meer dan 10 in stock zijn.

```
SELECT CategoryID, COUNT(productID) AS Aantal
FROM Products
WHERE UnitsInStock > 10
GROUP BY CategoryID
```



The screenshot shows a SQL query editor with the following query:

```
1 • SELECT CategoryID, COUNT(productID) AS Aantal
2 FROM Products
3 WHERE UnitsInStock > 10
4 GROUP BY CategoryID
```

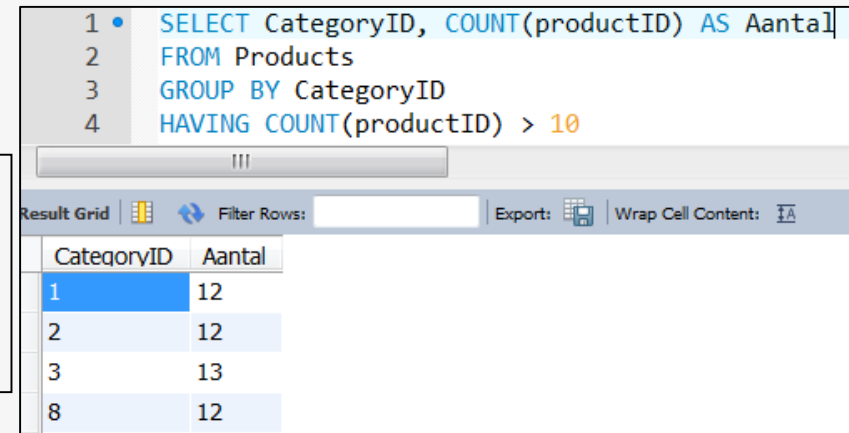
Below the query editor, there is a toolbar with buttons for 'Filter Rows', 'Export', and 'Wrap Cell Content'. Below the toolbar is a table with the following data:

CategoryID	Aantal
1	12
2	9
3	10
4	8
5	7
6	3
7	4
8	10

# Groeperen verfijnen via HAVING

- **HAVING** clause
  - Selecteren of verwerpen van groepen op basis van bepaalde groepeigenschappen
  - Enkele voorbeelden:
    - Toon per categorie die meer dan 10 producten bevat, het aantal producten

```
SELECT CategoryID, COUNT(productID) AS Aantal
FROM Products
GROUP BY CategoryID
HAVING COUNT(productID) > 10
```



The screenshot shows a SQL query editor with the following query:

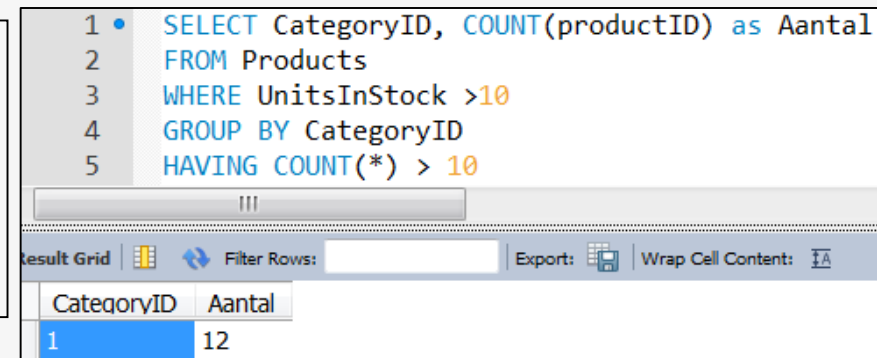
```
1 • SELECT CategoryID, COUNT(productID) AS Aantal
2 FROM Products
3 GROUP BY CategoryID
4 HAVING COUNT(productID) > 10
```

Below the query editor is a "Result Grid" with the following data:

CategoryID	Aantal
1	12
2	12
3	13
8	12

- Toon per categorie die meer dan 10 producten bevat, waarvan er meer dan 10 in stock zijn, het aantal producten

```
SELECT CategoryID, COUNT(productID) as Aantal
FROM Products
WHERE UnitsInStock >10
GROUP BY CategoryID
HAVING COUNT(*) > 10
```



The screenshot shows a SQL query editor with the following query:

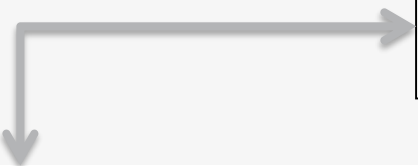
```
1 • SELECT CategoryID, COUNT(productID) as Aantal
2 FROM Products
3 WHERE UnitsInStock >10
4 GROUP BY CategoryID
5 HAVING COUNT(*) > 10
```

Below the query editor is a "Result Grid" with the following data:

CategoryID	Aantal
1	12

# WHERE vs HAVING

- Opmerkingen
  - Verschil tussen WHERE en HAVING
    - WHERE - heeft betrekking op rijen
    - HAVING - heeft betrekking op groepen
  - Statistische functies enkel gebruiken in SELECT, HAVING, ORDER BY niet in WHERE, GROUP BY
  - Indien er functies voorkomen in de select-clausule, dan moeten alle items van de SELECT-lijst, als argument van één of andere functie optreden **met uitzondering van de items van SELECT die voorkomen in de GROUP BY!!!**



```
SELECT categoryID, MIN(unitprice) AS Minimum  
FROM products
```

```
Server: Msg 8118, Level 16, State 1, Line 1  
Column 'products.CategoryID' is invalid in the select list because  
it is not contained in an aggregate function and there is no GROUP  
BY clause.
```

# Enkele oefeningen

NR	VNAAM	INIT	FNAAM	AFD	IN DIENST	CODE	NIV	GESL	GEBDAT	SALARIS
10	Christine	I	Haas	A00	650101	66	18	V	330814	52750
20	Michel	L	Theunis	B01	731001	61	18	M	480202	41250
30	Sally	A	Kramer	C01	750405	60	20	V	410511	38250
50	Johan	B	Geysen	E01	490817	58	16	M	250915	40175
60	Irving	F	Steur	D11	730914	55	16	M	450707	32250
70	Eva	D	Pulanski	D21	800930	56	16	V	530526	36170
90	Evelien	W	Hendriks	E11	700815	55	16	V	410515	29750
100	Theo	Q	Spencer	E21	800619	54	14	M	561218	26150
110	Vincent	G	Leman	A00	631205	58	19	M	291105	46500
120	Sean		Connors	A00	580516	58	14	M	421018	29250
130	Danielle	M	Scheire	C01	710728	55	16	V	250915	23800
140	Hilde	A	Nagels	C01	761215	56	18	V	460119	28420
150	Bruno		Adams	D11	720212	55	16	M	470517	25280
160	Els	R	Placke	D11	771011	54	17	V	550412	22250
170	Mats	J	Sierens	D11	780915	54	16	M	510105	24680
180	Marleen	S	Schouters	D11	730707	53	17	V	490221	21340
190	Jan	E	Wauters	D11	740726	53	16	M	520625	20450
200	David		De Bruyn	D11	660303	55	16	M	410529	27740
210	Willem	T	Jansens	D11	790411	25	17	M	530223	18270
220	Jennifer	K	Luyckx	D11	680829	55	18	V	480319	29840

AFDNR	AFDNAAM	MANNR
A00	Computer	10
B01	Planning	20
C01	Informatie	30
D01	Ontwikkelingssc	50
E01	Support	60
D11	Administratie	70
D21	Software	80
E21	Tools	90

tabel Afdeling

**HO  
GENT**

tabel Werknemer

# Enkele oefeningen

- Tel het aantal werknemers uit de afdeling D11 en geef het maximum, minimum en gemiddeld salaris voor deze afdeling, alsook het aantal verschillende jobcodes uit deze afdeling. Geef ook de som van alle lonen betaald in afdeling D11.
- Geef per afdeling, het afdnr en het aantal werknemers, gesorteerd volgens afdelingsnummer.
- Idem, maar nu gesorteerd volgens aantal werknemers.
- Idem maar nu wens je het aantal werknemers te kennen per afdeling en per jobcode.
- Tel per afdeling het aantal mannen en vrouwen en sorteer volgens opklimmende afdeling en afdalend geslacht.
- Geef een overzicht van de afdelingen die tenminste 2 werknemers hebben die meer dan 1000 verdienen.

A large, solid orange letter 'H' that serves as a background for the text 'Join.'.

Join.

**HO  
GENT**

# JOIN

- Selecteren van kolommen uit meerdere tabellen
  - JOIN keyword : specificeert de tabellen die samengevoegd moeten worden, en hoe ze moeten worden samengevoegd
    - Inner join
    - Outer join
    - Cross join
  - ON keyword : specificeert de JOIN voorwaarde
- Produceert 1 resultaatset, waarin de rijen uit die tabellen gekoppeld worden
- Basisvorm (ANSI JOIN (SQL-92) <-> Old style join)

```
SELECT uitdrukking  
FROM tabel JOIN tabel ON voorwaarde  
[JOIN tabel ON voorwaarde...]
```

```
SELECT uitdrukking  
FROM tabel, tabel [, tabel...]  
WHERE voorwaarde
```

# PUBS tabellen

- De voorbeelden maken gebruik van de volgende tabellen uit de DB Pubs





# INNER JOIN

- Koppelen van rijen uit één tabel met rijen uit een andere tabel op basis van gemeenschappelijke waarden in de overeenkomstige kolommen.
- De relatie tussen de velden in de verschillende tabellen kan je uitdrukken a.d.h.v.
  - = (equi-join)
  - <
  - >
  - <>
  - >=
  - <=

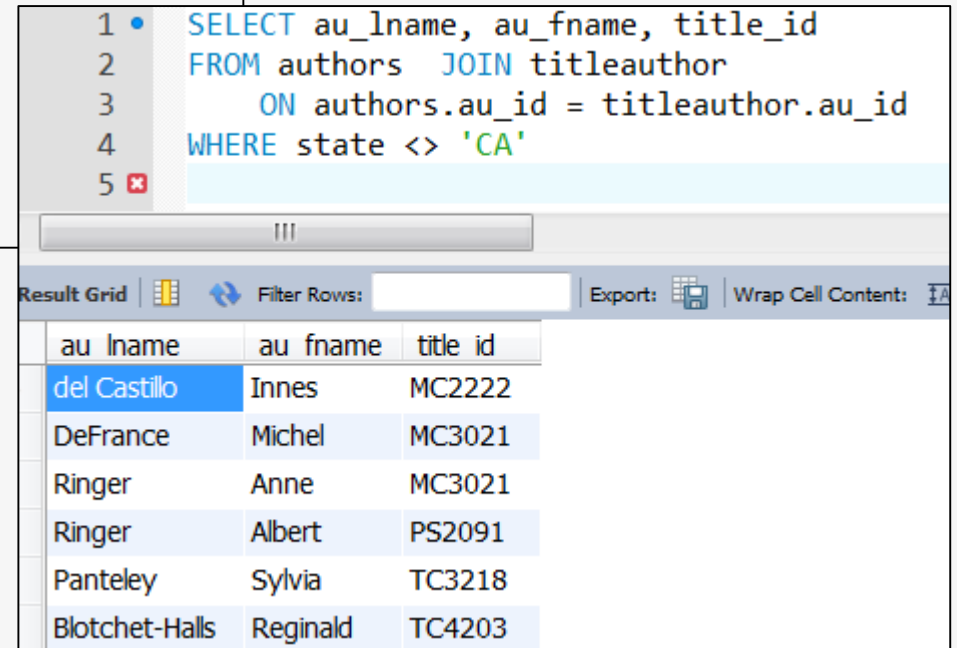
# INNER JOIN

- Voorbeeld van equi-join
  - Geef een overzicht van de auteurs (naam, voornaam) die niet in California wonen en de boeken (title\_id) die ze geschreven hebben.
    - ANSI JOIN (SQL-92)

```
SELECT au_lname, au_fname, title_id
FROM authors JOIN titleauthor
      ON authors.au_id = titleauthor.au_id
WHERE state <> 'CA'
```

- OF “old style join”

```
SELECT au_lname, au_fname, title_id
FROM authors, titleauthor
WHERE authors.au_id = titleauthor.au_id
      AND state <> 'CA'
```



The screenshot shows a database query editor with the following SQL query:

```
1 • SELECT au_lname, au_fname, title_id
2 FROM authors JOIN titleauthor
3     ON authors.au_id = titleauthor.au_id
4 WHERE state <> 'CA'
5 ✖
```

Below the query editor is a "Result Grid" showing the results of the query. The grid has three columns: "au\_lname", "au\_fname", and "title\_id". The results are as follows:

au_lname	au_fname	title_id
del Castillo	Innes	MC2222
DeFrance	Michel	MC3021
Ringer	Anne	MC3021
Ringer	Albert	PS2091
Panteley	Sylvia	TC3218
Blotchet-Halls	Reginald	TC4203

# Aliassen

- Gebruik van tabel aliasen (via 'AS' of spatie)

- SQL-92

```
SELECT au_lname, au_fname, title_id
FROM authors AS A
      JOIN titleauthor AS TA ON A.au_id = TA.au_id
WHERE state <> 'CA'
```

- “old style join”

```
SELECT au_lname, au_fname, title_id
FROM authors A, titleauthor TA
WHERE A.au_id = TA.au_id
      AND state <> 'CA'
```

## Opmerkingen:

1. Als een kolomnaam in meerdere tabellen (gebruikt in de query) voorkomt, dan **moet** die steeds worden voorafgegaan door de **tabelnaam (of alias)**.
2. Inner joins geven enkel die rijen terug die voldoen aan de ON conditie. Dit betekent dat als een rij in de eerste tabel niet matcht met een rij uit de tweede tabel (vb. een auteur die niet in California woont, en die nog geen boeken geschreven heeft) de rij niet zal geretourneerd worden en omgekeerd.
3. Als je in de old style join de where clause vergeet, dan krijg je de **cross join** (zie verder).

# INNER JOIN van meerdere tabellen

- JOIN van meer dan 2 tabellen
  - Voorbeeld: Geef een overzicht van de auteurs (naam, voornaam) die niet in California wonen en de boeken (titel) die ze geschreven hebben.
  - SQL-92:

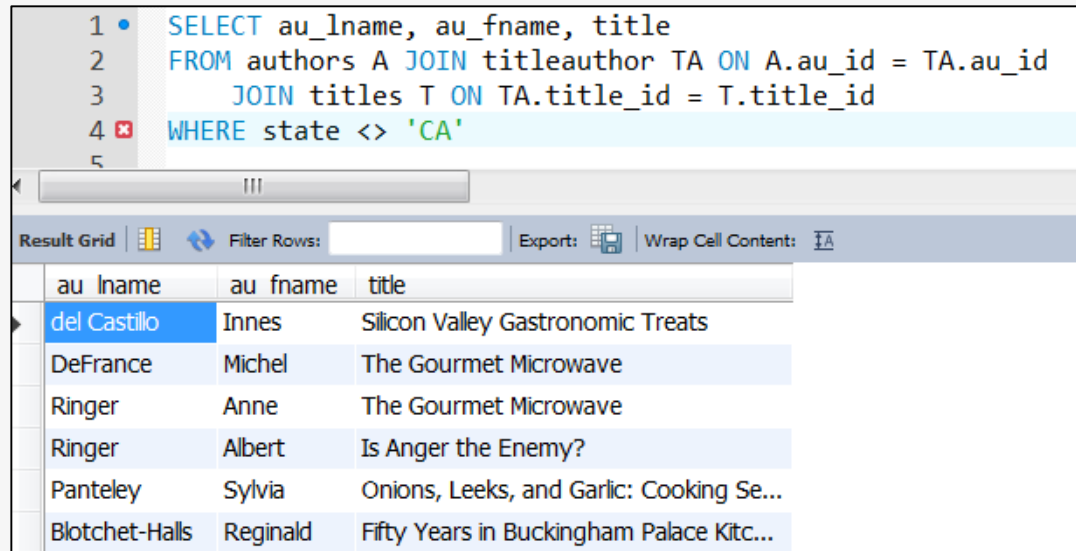
```
SELECT au_lname, au_fname, title
FROM authors A JOIN titleauthor TA ON A.au_id =
TA.au_id
      JOIN titles T ON TA.title_id = T.title_id
WHERE state <> 'CA'
```

Gegevens kunnen over meer dan 2 tabellen verspreid zitten.

Soms worden enkel gegevens uit 2 tabellen getoond, maar zijn toch extra tabellen nodig om de join te realiseren daar geen directe koppeling bestaat tussen de 2 tabellen waaruit de informatie moet komen.

- old style join

```
SELECT au_lname, au_fname, title
FROM authors A, titleauthor TA, titles T
WHERE A.au_id = TA.au_id
      AND TA.title_id = T.title_id
      AND state <> 'CA'
```



The screenshot shows a SQL query editor with the following query:

```
1 • SELECT au_lname, au_fname, title
2 FROM authors A JOIN titleauthor TA ON A.au_id = TA.au_id
3 JOIN titles T ON TA.title_id = T.title_id
4 WHERE state <> 'CA'
5
```

Below the query editor is a 'Result Grid' showing the results of the query. The grid has three columns: 'au\_lname', 'au\_fname', and 'title'. The results are as follows:

au_lname	au_fname	title
del Castillo	Innes	Silicon Valley Gastronomic Treats
DeFrance	Michel	The Gourmet Microwave
Ringer	Anne	The Gourmet Microwave
Ringer	Albert	Is Anger the Enemy?
Panteley	Sylvia	Onions, Leeks, and Garlic: Cooking Se...
Blotch-Halls	Reginald	Fifty Years in Buckingham Palace Kitc...

# INNER JOIN van een tabel met zichzelf

- Voorbeeld: Toon van alle werknemers het ID en de naam van hun manager

heeftAlsManager

employees
EmployeeID INT(11)
LastName VARCHAR(20)
FirstName VARCHAR(10)
Title VARCHAR(30)
TitleOfCourtesy VARCHAR(25)
BirthDate DATETIME
HireDate DATETIME
Address VARCHAR(60)
City VARCHAR(15)
Region VARCHAR(15)
PostalCode VARCHAR(10)
Country VARCHAR(15)
HomePhone VARCHAR(24)
Extension VARCHAR(4)
Photo LONGBLOB
Notes LONGTEXT
ReportsTo INT(11)
PhotoPath VARCHAR(255)
Indexes

```
1 SELECT X.employeeID as Employee, X.lastName, Y.employeeID as Manager, Y.lastName  
2 FROM employees as X JOIN employees as Y ON X.reportsTo= Y.employeeId
```

Employee	lastName	Manager	lastName
1	Davolio	2	Fuller
3	Leverling	2	Fuller
4	Peacock	2	Fuller
5	Buchanan	2	Fuller
6	Suyama	5	Buchanan
7	King	5	Buchanan
8	Callahan	2	Fuller
9	Dodsworth	5	Buchanan

vreemde sleutel

HO  
GENT

# OUTER JOIN

- Retourneert alle records van 1 tabel, zelfs als er geen gerelateerd record bestaat in de andere tabel.
- Er zijn 3 types van outer join
  - LEFT OUTER JOIN
    - retourneert alle rijen van de eerst genoemde tabel in de FROM clause (SQL-92)
  - RIGHT OUTER JOIN
    - retourneert alle rijen van de tweede tabel in de FROM clause (SQL-92)
  - FULL OUTER JOIN
    - retourneert ook rijen uit de eerste en tweede tabel die geen corresponderende entry hebben in andere tabel (SQL-92)

# LEFT OUTER JOIN

- Voorbeeld: Geef een overzicht van de auteurs (naam, voornaam), die niet wonen in California, en de boeken (titel) die ze geschreven hebben. Ook de auteurs die GEEN boeken geschreven hebben dienen op de lijst voor te komen.

```
SELECT au_lname, au_fname, title
FROM authors A
      LEFT JOIN titleauthor TA ON A.au_id = TA.au_id
      LEFT JOIN titles T ON TA.title_id = T.title_id
WHERE state <> 'CA'
```

```
1 • SELECT au_lname, au_fname, title
2 FROM authors A
3     LEFT JOIN titleauthor TA ON A.au_id = TA.au_id
4     LEFT JOIN titles T ON TA.title_id = T.title_id
5 WHERE state <> 'CA'
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

au_lname	au_fname	title
Smith	Meander	NULL
Greene	Morningstar	NULL
Blotch-Halls	Reginald	Fifty Years in Buckingham Palace Kitc...
del Castillo	Innes	Silicon Valley Gastronomic Treats
DeFrance	Michel	The Gourmet Microwave
Panteley	Sylvia	Onions, Leeks, and Garlic: Cooking Se...
Ringer	Anne	The Gourmet Microwave
Ringer	Albert	Is Anger the Enemy?

# RIGHT OUTER JOIN

- Voorbeeld: Toon een lijst van de boeken, met naam en voornaam van de auteurs. Enkel de boeken geschreven door auteurs die niet in CA wonen of de boeken waarvoor de auteur niet bekend is mogen op het overzicht voorkomen.

```
SELECT au_lname, au_fname, title
FROM authors AS A
      RIGHT JOIN titleauthor AS TA ON A.au_id = TA.au_id
      RIGHT JOIN titles AS T ON TA.title_id = T.title_id
WHERE state <> 'CA' OR A.state IS NULL
```

au_lname	au_fname	title
del Castillo	Innes	Silicon Valley Gastronomic Treats
DeFrance	Michel	The Gourmet Microwave
Ringer	Anne	The Gourmet Microwave
Ringer	Albert	Is Anger the Enemy?
NULL	NULL	Life Without Fear
NULL	NULL	Emotional Security: A New Algorithm
Panteley	Sylvia	Onions, Leeks, and Garlic: Cooking Se...
Blotchett-Halls	Reginald	Fifty Years in Buckingham Palace Kitc...



# VOORBEELD

T1

C1	C2
1	A
1	B
2	B

T2

k1	k2
1	X
1	Y
3	Y

Select \* from T1 join T2 on T1.c1=T2.k1

T1.C1	T1.C2	T2.K1	T2.K2
1	A	1	X
1	A	1	Y
1	B	1	X
1	B	1	Y

Select \* from T1 left join T2 on T1.c1=T2.k1

T1.C1	T1.C2	T2.K1	T2.K2
1	A	1	X
1	A	1	Y
1	B	1	X
1	B	1	Y
2	B	NULL	NULL

Select \* from T1 right join T2 on T1.c1=T2.k1

T1.C1	T1.C2	T2.K1	T2.K2
1	A	1	X
1	B	1	X
1	A	1	Y
1	B	1	Y
NULL	NULL	3	Y

Select \* from T1 full outer join T2 on T1.c1=T2.k1

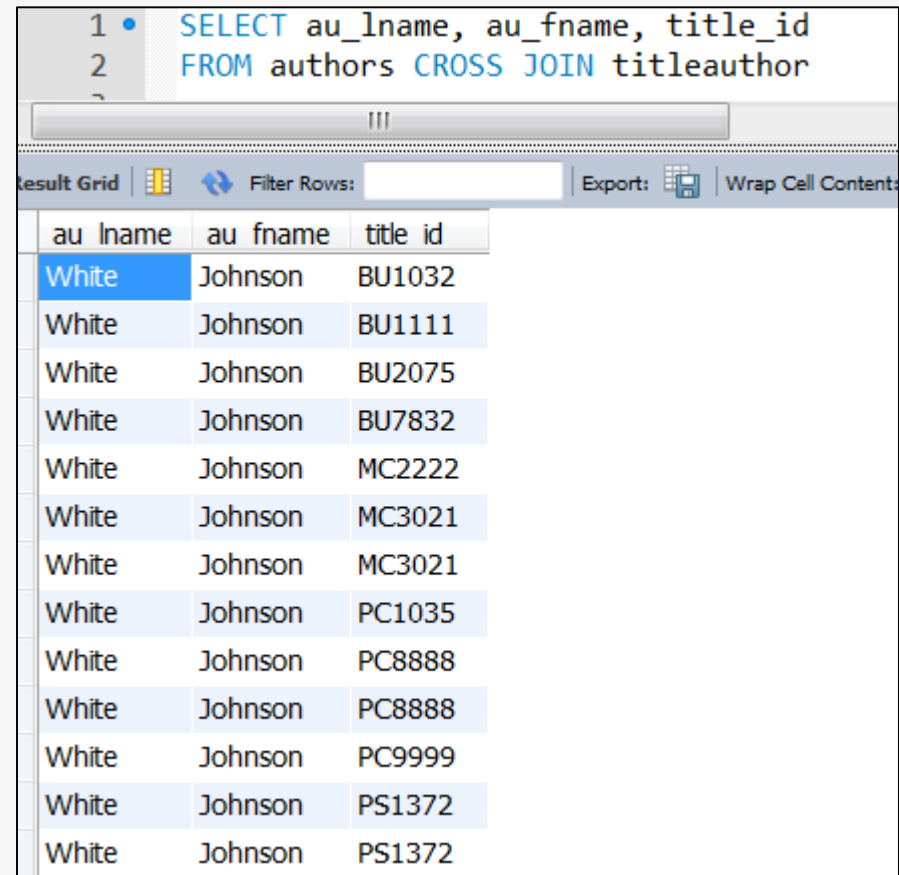
T1.C1	T1.C2	T2.K1	T2.K2
1	A	1	X
1	A	1	Y
1	B	1	X
1	B	1	Y
2	B	NULL	NULL
NULL	NULL	3	Y

# CROSS JOIN

- Bij een cross join is het aantal rijen in de resultaat tabel gelijk aan het aantal rijen in de eerste tabel maal het aantal rijen in de tweede tabel.
  - SQL-92

```
SELECT au_lname, au_fname, title_id
FROM authors CROSS JOIN titleauthor
```
  - old style join

```
SELECT au_lname, au_fname, title_id
FROM authors, titleauthor
```



The screenshot shows a SQL query editor with a query window at the top and a result grid below it. The query window contains the following SQL statement:

```
1 • SELECT au_lname, au_fname, title_id
2 FROM authors CROSS JOIN titleauthor
3
```

The result grid displays the output of the query, showing a Cartesian product of the authors and titleauthor tables. The columns are labeled 'au\_lname', 'au\_fname', and 'title\_id'. The data is as follows:

au_lname	au_fname	title_id
White	Johnson	BU1032
White	Johnson	BU1111
White	Johnson	BU2075
White	Johnson	BU7832
White	Johnson	MC2222
White	Johnson	MC3021
White	Johnson	MC3021
White	Johnson	PC1035
White	Johnson	PC8888
White	Johnson	PC8888
White	Johnson	PC9999
White	Johnson	PS1372
White	Johnson	PS1372

A large, solid orange letter 'H' that serves as a background for the text 'Union.'. The letter is composed of two vertical bars and a horizontal crossbar, all in a uniform orange color.

Union.

**HO  
GENT**

# UNION

- Via een **UNION** combineer je het resultaat van 2 of meerdere queries in 1 resultaat tabel

- Basisvorm

```
SELECT ... FROM ... WHERE ...  
UNION  
SELECT ... FROM ... WHERE ...  
ORDER BY ...
```

- Regels

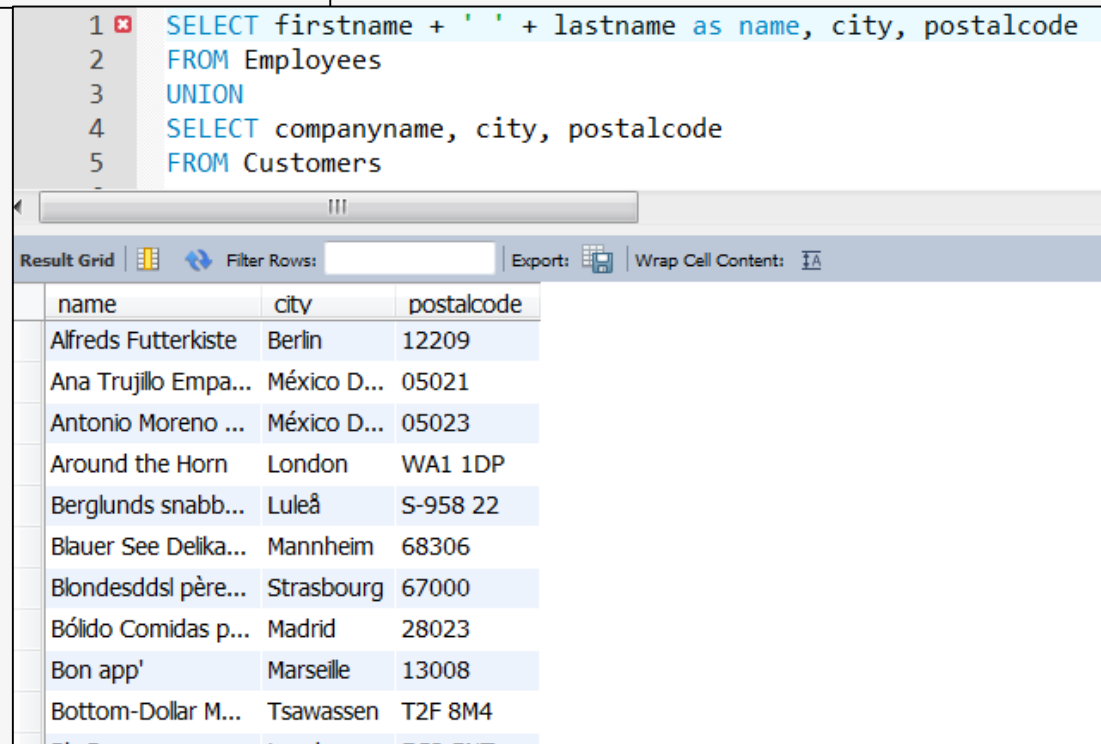
- De resultaten van de 2 SELECT opdrachten moeten evenveel kolommen bevatten.
- Overeenkomstige kolommen uit beide SELECT's moeten van hetzelfde data type zijn en beide NOT NULL toelaten of niet.
- Kolommen komen voor in dezelfde volgorde.
- De kolomnamen/titels van de UNION zijn deze van de eerste SELECT.
- Het resultaat bevat echter steeds alleen unieke rijen.
- Aan het einde van de UNION kan je een ORDER BY toevoegen. In deze clausule mag geen kolomnaam of uitdrukking voorkomen indien kolomnamen van beide select's verschillend zijn. Gebruik in dat geval kolomnummers.

# UNION

- Voorbeeld: geef een overzicht van alle bedienden (naam en voornaam, stad en postcode) en alle klanten (naam, stad en postcode)

```
SELECT firstname + ' ' + lastname as name, city, postalcode
FROM Employees
UNION
SELECT companyname, city, postalcode
FROM Customers
```

*Daar de kolomnamen van de resultaatset van de UNION deze zijn van de eerste select, dien je de titel 'name' in de tweede select niet meer te herhalen.*



The screenshot shows a SQL query editor with a query window and a result grid. The query is as follows:

```
1 SELECT firstname + ' ' + lastname as name, city, postalcode
2 FROM Employees
3 UNION
4 SELECT companyname, city, postalcode
5 FROM Customers
```

The result grid displays the following data:

name	city	postalcode
Alfreds Futterkiste	Berlin	12209
Ana Trujillo Empa...	México D...	05021
Antonio Moreno ...	México D...	05023
Around the Horn	London	WA1 1DP
Berglunds snabb...	Luleå	S-958 22
Blauer See Delika...	Mannheim	68306
Blondesddsl père...	Strasbourg	67000
Bóldo Comidas p...	Madrid	28023
Bon app'	Marseille	13008
Bottom-Dollar M...	Tsawassen	T2F 8M4
Bla Bla Bla...	London	EC2 5NT