

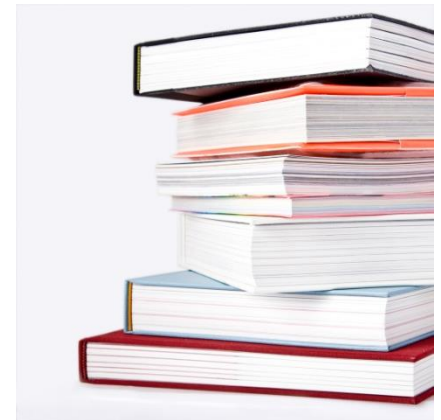
Hoofdstuk 5

Analytische functies

(= aggregatie functies,
statistische functies)

cursusmateriaal

- Handboek 'SQL voor hoger onderwijs' blz. 120
- SQL Reference blz 9-24
- Deze powerpoint



Hoofdstuk 5: Analytische functies

De database  informatiebron voor een organisatie


op basis van die
informatie worden
beslissingen genomen

vaak informatie nodig zoals:
gemiddelden
aantallen
totalen
maxima/minima

Om die informatie uit de database te halen moet je gegevens groeperen en **analytische functies** gebruiken

Hoofdstuk 5: Analytische functies

COUNT(*)

COUNT([ALL|DISTINCT] *expressie*)

SUM ([ALL|DISTINCT] *expressie*)

AVG ([ALL|DISTINCT] *expressie*)

MIN(*expressie*)

MAX(*expressie*)

waarbij “*expressie*” kan zijn :

- kolomnaam
- constante
- functie
- combinatie van kolomnamen, functies, constanten gekoppeld door wiskundige operatoren

Hoofdstuk 5: Analytische functies

- een statistische functie geeft per groep **1 RIJ RESULTAAT**
- statistische functies staan **steeds in de SELECT list** (of in de **HAVING clause**)
statistische functies staan **dus NOOIT in de WHERE clause!!!!**
- statistische functies **houden geen rekening met NULL waarden**
- **in de SELECT list** kunnen er **naast analytische functies** enkel nog **constanten** voorkomen

Hoofdstuk 5: Analytische functies

COUNT(*)

COUNT([ALL|DISTINCT] *expressie*)

SUM ([ALL|DISTINCT] *expressie*)

AVG ([ALL|DISTINCT] *expressie*)

MIN(*expressie*)

MAX(*expressie*)

AVG = Average

AVG ([ALL|DISTINCT] *expressie*)

berekent het **gemiddelde** van **numerieke** gegevens

Wat is het gemiddelde salaris van het bedrijf?

```
SELECT AVG(salaris) "GEMIDDELD SALARIS"  
  
FROM medewerkers;
```

```
GEMIDDELD SALARIS  
-----  
35500
```



MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Geeft 1 rij als resultaat.

Wat is het gemiddelde van de verschillende salarissen van het bedrijf?

```
SELECT AVG(DISTINCT salaris) "GEMIDDELD SALARIS"  
  
FROM medewerkers;
```

```
GEMIDDELD SALARIS  
-----  
38200
```

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Hoofdstuk 5: Analytische functies

COUNT(*)

COUNT([ALL|DISTINCT] *expressie*)

SUM ([ALL|DISTINCT] *expressie*)

AVG ([ALL|DISTINCT] *expressie*)

MIN(*expressie*)

MAX(*expressie*)

SUM = Som


SUM([ALL|DISTINCT] *expressie*)

berekent de **som** van **numerieke** waarden

Wat is de totale jaarlijkse loonkost van het bedrijf?

```
SELECT SUM(salaris) totaal  
FROM medewerkers;
```

```
TOTAAL  
-----  
284000
```



MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Geeft 1 rij als resultaat.

Wat is het totaal van de verschillende salarissen?

```
SELECT SUM(DISTINCT salaris) totaal  
FROM medewerkers;
```

```
TOTAAL  
-----  
191000
```

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Hoofdstuk 5: Analytische functies

COUNT(*)

COUNT([ALL|DISTINCT] *expressie*)

SUM ([ALL|DISTINCT] *expressie*)

AVG ([ALL|DISTINCT] *expressie*)

MIN(*expressie*)

MAX(*expressie*)

MAX en MIN

MIN(expressie)

MAX(expressie)

berekent het **maximum** resp. **minimum** van attribuutwaarden

De functie kan op elk type attribuut toegepast worden.

Wat is het hoogste en laagste salaris van het bedrijf?

```
SELECT MAX(salaris) "hoogste salaris",  
MIN(salaris) "laagste salaris"  
FROM medewerkers;
```

```
hoogste salaris laagste salaris  
-----  
55000 25000
```

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Welke achternaam komt eerst/laatst in het alfabet?

```
SELECT  MIN(achternaam) "eerste",  
        MAX(achternaam) "laatste"  
FROM medewerkers;
```

eerste	laatste
-----	-----
Amelsvoort	Zuiderweg

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Wat is de geboortedatum van de oudste/jongste medewerker?

```
SELECT  MIN(geb_datum) "oudste",  
        MAX(geb_datum) "jongste"  
FROM medewerkers;
```

```
oudste  jongste  
-----  
01/09/65 19/07/88
```

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Hoofdstuk 5: Analytische functies

COUNT(*)

COUNT([ALL|DISTINCT] *expressie*)

SUM ([ALL|DISTINCT] *expressie*)

AVG ([ALL|DISTINCT] *expressie*)

MIN(*expressie*)

MAX(*expressie*)

COUNT

COUNT(*)

telt geselecteerde rijen

COUNT([ALL|DISTINCT] *expressie*)

telt attribuutwaarden

Kan op elk type attribuut toegepast worden.

Hoeveel medewerkers telt het bedrijf?

```
SELECT COUNT(*) aantal  
FROM medewerkers;
```

```
AANTAL  
-----  
      8
```

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

COUNT(*) telt geselecteerde rijen
(als er geen rijen geselecteerd
worden is count(*)=0)

Hoeveel medewerkers hebben een baas?

```
SELECT sofi_nr, achternaam, mgr_sofi_nr  
FROM medewerkers;
```

SOFI_NR	ACHTERNAAM	MGR_SOFI_NR
999666666	Bordoloi	
999555555	Jochems	999666666
999444444	Zuiderweg	999666666
999887777	Muiden	999555555
999222222	Amelsvoort	999555555
999111111	Bock	999444444
999333333	Joosten	999444444
999888888	Pregers	999444444

8 rows selected

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Hoeveel medewerkers hebben een baas?

```
SELECT COUNT(mgr_sofi_nr) aantal  
FROM medewerkers;
```

```
AANTAL  
-----  
          7
```

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

COUNT(([ALL|DISTINCT] *expressie*) -> je geeft een specifieke kolom of expressie op voor de telling en je telt geldige attribuutwaarden (enkel de ingevulde waarden!).

Je kan analytische functies ook toepassen op een aantal geselecteerde rijen.

```
SELECT AVG(salaris) gem_sal_afd3
FROM medewerkers
WHERE afd_nr=3;
```

```
GEM_SAL_AFD3
-----
          31000
```

```
SELECT SUM(salaris) tot_sal_afd7
FROM medewerkers
WHERE afd_nr=7;
```

```
TOT_SAL_AFD7
-----
        136000
```

```
SELECT  MIN(achternaam) "eerst in alfabet",
        MAX(achternaam) "laatst in alfabet"
FROM medewerkers
WHERE afd_nr=3;
```

eerst in alfabet	laatst in alfabet
Amelsvoort	Muiden

```
SELECT COUNT(*) aantal_afd3
FROM medewerkers
WHERE afd_nr=3;
```

```
AANTAL_AFD3
-----
          3
```


Hoofdstuk 5

Group By

Hoofdstuk 5: Group By

Tot nu toe:

Analytische functies toegepast op:

ALLE rijen uit de tabel

of

een AANTAL GESELECTEERDE rijen

uit de tabel

Maar:

- Je kan rijen ook groeperen volgens een bepaald criterium en de analytische functies per groepje uitvoeren

Hoofdstuk 5: Group By - voorbeelden

- het aantal medewerkers **per afdeling** tellen
 - medewerkers per afdeling groeperen + hun aantal per groep tellen
- de totale loonkost **per afdeling** berekenen
 - alle medewerkers per afdeling groeperen en per groep de totale loonkost berekenen
- het gemiddeld aantal bestede uren **per project** berekenen ...
 - alle rijen betreffende hetzelfde project groeperen en per gevormde groep het gemiddeld aantal uren berekenen

Geef het aantal medewerkers per afdeling

Oplossing 1

```
SELECT sofi_nr,voornaam,achternaam,afd_nr  
FROM medewerkers;
```

SOFI_NR	VOORNAAM	ACHTERNAAM	AFD_NR
999666666	Bijoy	Bordoloi	1
999555555	Suzan	Jochems	3
999444444	Willem	Zuiderweg	7
999887777	Martina	Muiden	3
999222222	Henk	Amelsvoort	3
999111111	Douglas	Bock	7
999333333	Dennis	Joosten	7
999888888	Shanya	Pregers	7

8 rows selected

MEDEWERKERS	
P *	SOFI_NR CHAR (9 BYTE)
*	ACHTERNAAM VARCHAR2 (25 BYTE)
*	VOORNAAM VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL VARCHAR2 (25 BYTE)
	ADRES VARCHAR2 (50 BYTE)
	PLAATS VARCHAR2 (25 BYTE)
	PROVINCIE CHAR (2 BYTE)
	POSTCODE VARCHAR2 (7 BYTE)
	GEB_DATUM DATE
	SALARIS NUMBER (7,2)
U	PARKEERPLAATS NUMBER (4)
	GESLACHT CHAR (1 BYTE)
F	AFD_NR NUMBER (2)
F	MGR_SOFI_NR CHAR (9 BYTE)

Bepaal het aantal medewerkers per afdeling!

```
SELECT  COUNT(*) "aantal afd 1"
FROM    medewerkers
WHERE   afd_nr=1;

aantal afd 1
-----
1
```

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Je doet dit voor Afdeling 1, daarna voor Afdeling 3 enz...

Bepaal het aantal medewerkers per afdeling!

```
SELECT  COUNT(*) "aantal afd 1"  
  FROM medewerkers  
 WHERE afd_nr=1;
```

```
aantal afd 1  
-----  
1
```

```
SELECT  COUNT(*) "aantal afd 3"  
  FROM medewerkers  
 WHERE afd_nr=3;
```

```
aantal afd 3  
-----  
3
```

```
SELECT  COUNT(*) "aantal afd 7"  
  FROM medewerkers  
 WHERE afd_nr=7;
```

```
aantal afd 7  
-----  
4
```

Enz....

Vraag: Wat als er 10 afdelingen zijn?

Vraag: Wat als er een afdeling zou bijkomen?

Vraag: Wat als het afd_nr verandert?

Conclusie: Deze OPLOSSING is niet efficiënt

Bepaal het aantal medewerkers per afdeling!

```
SELECT afd_nr, COUNT(*) aantal
FROM medewerkers
GROUP BY afd_nr;
```

AFD_NR	AANTAL
1	1
3	3
7	4

MEDEWERKERS		
P	* SOFI_NR	CHAR (9 BYTE)
	* ACHTERNAAM	VARCHAR2 (25 BYTE)
	* VOORNAAM	VARCHAR2 (25 BYTE)
	TUSSENVOEGSEL	VARCHAR2 (25 BYTE)
	ADRES	VARCHAR2 (50 BYTE)
	PLAATS	VARCHAR2 (25 BYTE)
	PROVINCIE	CHAR (2 BYTE)
	POSTCODE	VARCHAR2 (7 BYTE)
	GEB_DATUM	DATE
	SALARIS	NUMBER (7,2)
U	PARKEERPLAATS	NUMBER (4)
	GESLACHT	CHAR (1 BYTE)
F	AFD_NR	NUMBER (2)
F	MGR_SOFI_NR	CHAR (9 BYTE)

Bepaal het aantal medewerkers per afdeling!

```
SELECT afd_nr, COUNT(*)  aantal      3  
FROM medewerkers        1  
GROUP BY afd_nr;        2
```

1. de tabel in de FROM lijn wordt aangesproken
2. de rijen worden gegroepeerd op basis van de groepskenmerken (dit zijn de attributen vermeld achter GROUP BY) -> alle rijen betreffende eenzelfde afdeling komen in hetzelfde groepje te staan
3. voor ELK groepje wordt de select list uitgevoerd en dit levert per groepje één rij resultaat.


```

SELECT afd_nr,COUNT(*)  aantal      3
FROM medewerkers        1
GROUP BY afd_nr;        2

```

1. de tabel in de FROM lijn wordt aangesproken
2. de rijen worden gegroepeerd op basis van de groepskenmerken
(dit zijn de attributen vermeld achter GROUP BY) -> alle rijen
betreffende eenzelfde afdeling komen in hetzelfde groepje te staan
3. voor ELK groepje wordt de select list uitgevoerd en dit levert per groepje één rij
resultaat.

```

1 ----- }
3 ----- }
3 ----- }
3 ----- }
7 ----- }
7 ----- }
7 ----- }
7 ----- }

```

AFD_NR	AANTAL
-----	-----
1	1
3	3
7	4

Group BY

Bij GROUP BY zijn in de SELECT list toegelaten:

- analytische functies
- constanten
- attributen die achter GROUP BY staan
(=groepskenmerken)

Hoofdstuk 5: Group By

- In combinatie met WHERE

Bereken per afdeling de jaarlijkse loonkost voor vrouwen!

```
SELECT afd_nr, SUM(salaris) "jaarlijkse loonkost"  
FROM medewerkers  
WHERE LOWER(geslacht)='v'  
GROUP BY afd_nr;
```

1. de tabel in de FROM lijn wordt aangesproken
2. er wordt bepaald welke rijen meespelen
3. die rijen worden gegroepeerd op basis van afd_nr
4. voor elke gevormde groep wordt de SELECT list uitgevoerd
→dit levert 1 rij op per groep!

AFD_NR	jaarlijkse loonkost
3	68000
7	25000

Hoofdstuk 5: Group By

- In combinatie met ORDER BY

Bereken per afdeling de gemiddelde loonkost! Sorteer in stijgende volgorde van gemiddelde

```
SELECT afd_nr,AVG(salaris) gemiddeld_sal  
FROM medewerkers  
GROUP BY afd_nr  
ORDER BY 2;
```

1. de tabel in de FROM lijn wordt aangesproken
2. de rijen worden gegroepeerd op basis van afd_nr
3. select list wordt uitgevoerd
4. order by wordt uitgevoerd

AFD_NR	GEMIDDELD_SAL
3	31000
7	34000
1	55000

Bereken per afdeling de totale loonkost voor mannen ! Sorteer in dalende volgorde van loonkost

```
SELECT  afd_nr, SUM(salaris)  TOT_LOONKOST
FROM    medewerkers
WHERE   UPPER(geslacht)='M'
GROUP BY afd_nr
ORDER BY 2 DESC;
```

AFD_NR	TOT_LOONKOST
7	111000
1	55000
3	25000

Nesten van analytische functies

- Analytische functie toegepast op een groep rijen geeft 1 rij resultaat
 - geen nesten van analytische functies mogelijk
(m.a.w. op dat resultaat kan je geen analytische functie meer uitvoeren)
- Analytische functies in combinatie met GROUP BY geven per gevormd groepje een rij resultaat
 - op die rijen kan opnieuw een analytische functie uitgevoerd worden

Nesten van analytische functies

- Voorbeeld: Geef het hoogste salaris

```
SELECT achternaam, salaris
FROM medewerkers;
```

ACHTERNAAM	SALARIS
Bordoloi	55000
Jochems	43000
Zuiderweg	43000
Muiden	25000
Amelsvoort	25000
Bock	30000
Joosten	38000
Pregers	25000

```
SELECT MAX(salaris) max_sal
FROM medewerkers;
```

MAX_SAL
55000

8 rows selected

Nesten van analytische functies

Geef het gemiddelde van het maximum salaris.

```
SELECT AVG (MAX (salaris))  
FROM medewerkers;
```

zinloos

syntax fout



ORA-00978: nested group function without GROUP BY

Geef per afdeling het hoogste salaris

```
SELECT afd_nr, MAX(salaris) maximum  
FROM medewerkers  
GROUP BY afd_nr;
```

AFD_NR	MAXIMUM
1	55000
3	43000
7	43000

je kan op de verzameling van deze 3 rijen opnieuw een analytische functie uitvoeren!



in combinatie met GROUP BY
kunnen functies wel genest worden!

Geef het gemiddelde van de hoogste salarissen per afdeling

```
SELECT afd_nr, AVG(MAX(salaris)) "gem. van hoogste per afdeling"  
FROM medewerkers  
GROUP BY afd_nr;
```



ORA-00937: not a single-group group function

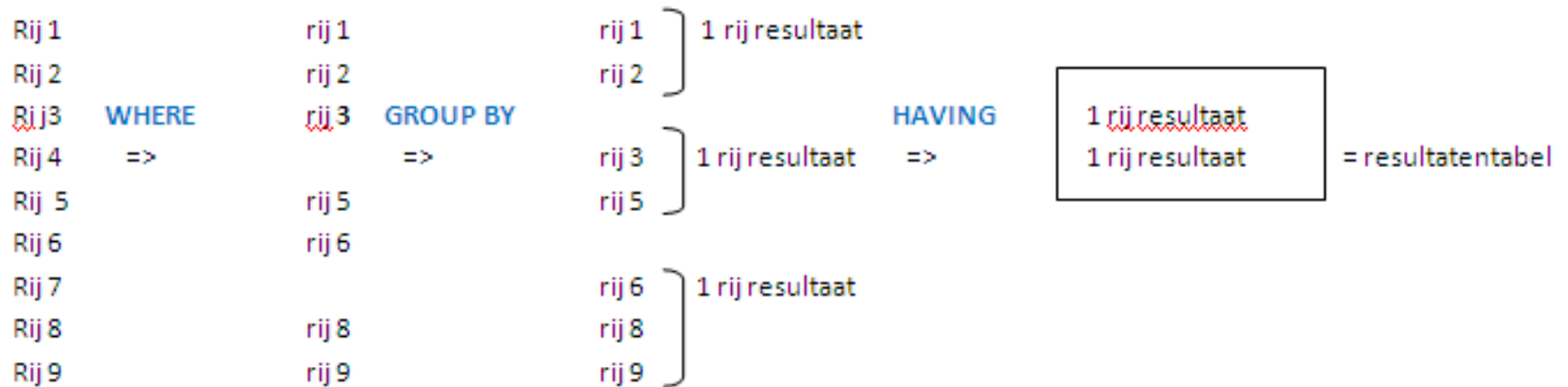
```
gem. van hoogste per afdeling  
-----  
47000
```

Geneste analytische functies kan je in de SELECT list niet combineren met attributen, wel met constanten en andere geneste functies

Hoofdstuk 5: Group By de HAVING Clausule

De HAVING clausule doet met de uit GROUP BY bekomen rijen wat de WHERE clausule doet met rijen uit de tabel

HAVING Clause



Geef een overzicht van de afdelingen met een gemiddeld salaris dat boven 33000 gelegen is

```
SELECT afd_nr, AVG(salaris)  
FROM medewerkers  
WHERE AVG(salaris) > 33000;
```

Waarom is deze oplossing fout?

```
SELECT afd_nr, AVG(salaris) gem_sal  
FROM medewerkers  
GROUP BY afd_nr;
```

AFD_NR	GEM_SAL
1	55000
3	31000
7	34000

Geef een overzicht van de afdelingen met een gemiddeld salaris dat boven 33000 gelegen is

```
SELECT afd_nr,AVG(salaris)
FROM medewerkers
GROUP BY afd_nr
HAVING AVG(salaris)>33000;
```

AFD_NR	AVG(SALARIS)
1	55000
7	34000

HAVING behoudt uit de uit GROUP BY bekomen rijen enkel die rijen die voldoen aan de voorwaarde vermeld achter HAVING.

Geef per afdeling het gemiddelde salaris. We zijn alleen geïnteresseerd in afdelingen met meer dan 2 medewerkers

```
SELECT afd_nr, AVG(salaris) gemiddeld_sal  
FROM medewerkers  
GROUP BY afd_nr  
HAVING COUNT(*) > 2;
```

AFD_NR	GEMIDDELD_SAL
3	31000
7	34000

Je kan in de HAVING clause op een andere analytische functie testen dan de analytische functie die je gebruikt in de SELECT-list

Wat mag je in de HAVING clause plaatsen?

- Eender welke analytische functies op attributen uit de tabel in de FROM lijn
- attributen uit de groepskenmerken