MODULE DATABASES
CSG AUGUSTINUS

GEGEVENSBANKEN, SQL & DATASTRUCTUREN

INHOUDSOPGAVE

Inhoud

INLEIDING

FIGUUR 1

Ken je iemand wiens bureau eruit ziet zoals in figuur 1? De berg papieren bevat een hoop gegevens of **data**, maar ze leveren de eigenaar van het bureau op dit moment misschien niet de gewenste **informatie**.

De noodzaak om data te structureren wordt groter, naarmate je meer data hebt. Dat geldt in jouw eigen omgeving, maar ook in de digitale wereld. Wie de data niet op een herkenbare manier ordent, heeft geen overzicht. Je kunt er dan geen **informatie** uithalen of nieuwe **kennis** mee vergaren.

Als je de data wel ordent, kun je er **informatie** uithalen: de data krijgt betekenis. Als je gegevens met elkaar weet te verbinden of er patronen in ontdekt, ontwikkel je nieuwe **kennis**. Zo levert één weerstation **data** (in de vorm van de spanning van de weersensoren) over de lokale weersituatie die wordt omgezet naar **informatie** over bijvoorbeeld windkracht, temperatuur en luchtdruk. Meerdere weerstations leveren een totaalbeeld voor een land, dat kan bijdragen aan de **kennis** in de vorm van een weerbericht.

Bij de lessen over programmeren heb je kennis gemaakt met het begrip **variabele**. In een variabele kan een gegeven worden opgeslagen onder een bepaalde naam, zoals *huisnummer = 4*. Een bijzondere variabele was de **array**, waarin je een reeks met gegevens onder één naam kon bewaren. Een voorbeeld hiervan is: *onderwerpVanHetHoofdstuk = ['html','javascript,'hardware','databases']*.
Als we dit idee verder uitbreiden komen we uit bij een **tabel**. Op de volgende pagina staan twee tabellen die in het vervolg van dit hoofdstuk vaker zullen worden gebruikt. De tabellen horen bij elkaar. We zeggen dat ze een **relatie** hebben. Een verzameling tabellen die een relatie met elkaar hebben, heet een **relationele database**. Het werken met databases is het hoofdonderwerp van deze module.

Opdracht 1: data, informatie en kennis

1. Wat is het verschil tussen data en informatie? Gebruik de theorie hierboven en internet.
2. In de theorie wordt gesproken over data, informatie en kennis. In deze volgorde neemt de waarde van de gegevens steeds verder toe. Zoek op internet op wat de vierde stap is.
3. Om een voorspelling van het weer van volgende week te kunnen doen, wordt een grote hoeveelheid data, informatie en kennis gecombineerd.
   Bedenk twee andere voorbeelden waarbij dit ook het geval is.
4. *Big data* is een actueel item binnen de ICT. Zoek op wat de term betekent.
5. Noem drie bedrijven die gegevens van jou hebben.
6. Bedenk een bedrijf dat iets zou kunnen hebben aan die gegevens. Leg uit waar ze de gegevens voor kunnen gebruiken.
7. Stel dat je een ruim budget hebt om gegevens te kopen. Welke gegevens zou je graag willen hebben? En waarom?

Opdracht 2: informatie uit tabellen halen

Bekijk de tekst op de pagina hiernaast en bestudeer de tabellen. Beantwoord vervolgens de vragen.

8. Van welk type meter bestaan de meeste varianten in de tabel?
9. Welk type meter is het vaakst vervangen? Hoe vaak?
10. Geef het type en het merk van de meter die op dit moment het meest in gebruik is bij de weerstations.
11. Welke meter werd het eerst van allemaal vervangen? En welke werd het snelst vervangen? Geef het type en het merk en het weerstation waar deze vervanging plaats had.
12. Hoeveel meters die werken op een spanning van 12V worden op dit moment nog gebruikt?
13. Welke beheerders maken op dit moment gebruik van meters die werken op 3V?
14. Bedenk drie vragen die je over deze tabellen zou kunnen stellen. Laat je buurman antwoorden.

Database Weerstations

Hiernaast zie je een tabel met samenwerkende weerstations in de provincie Groningen. Deze tabel **stations** spreekt waarschijnlijk voor zich: hierin staan de verschillende weerstations op een rij met de plaats waar ze zich bevinden en de persoon die het lokale weerstation beheert.

stations		
ws_id	plaats	beheerder
1	Zuidhorn	Van der Veen
2	Groningen	Velthuizen
3	Groningen	Grgurina
4	Bedum	Palsma
5	Garrelsweer	Osinga

FIGUUR 2

Een weerstation bevat verschillende **meters**. Er is een tabel met gegevens over verschillende typen meters. Hierbij staat ook vermeld van welk merk ze zijn en op welke spanning (voeding) ze werken. Voor elk type meter is er een unieke identificatiecode (*m_id*) gemaakt.

Niet elk weerstation zet dezelfde meters in. Beheerders maken zelf een keuze. Bovendien zijn in de loop der jaren meters kapot gegaan en vervangen door een ander type. Informatie hierover is te vinden in de tabel **inzet**.
Van een bepaald type meter kunnen meerdere exemplaren worden ingezet. Alle exemplaren hebben hun eigen metercode (*m_code*). Dit betekent dat bij één *m_id* meerdere metercodes (*m_code*) kunnen horen. Wanneer een meter is vervangen, is dat te zien doordat in de tabel het veld *eind* is gevuld. In s het veld niet gevuld, dan krijgt deze in de database de waarde **NULL**. De kolom *start* geeft de plaatsingsdatum van een nieuwe meter.

De tabel **inzet** is het lastigst te lezen en te begrijpen. De eerste regel met metercode (*m_code*) 3681 heeft 3 als *ws_id*. Dit is een identificatiecode van het weerstation van beheerder Grgurina uit de tabel **weerstations**.
De meter met de unieke code 3681 heeft metercode (*m_code*) 2. Dit betekent dat het een luchtdrukmeter van het merk Samsung is met een voeding van 12V. Dit blijkt uit de tabel **meters**.

Uit dit voorbeeld blijkt dat deze drie tabellen een relatie met elkaar hebben. Samen vormen de drie tabellen een gegevensbank of **relationele database**.

meters			
m_id	type	merk	voeding
1	luchtdruk	Samsung	5V
2	luchtdruk	Samsung	12V
3	luchtdruk	Philips	5V
4	luchtdruk	Philips	5V
5	luchtdruk	Vavetech	5V
6	luchtdruk	Vavetech	12V
7	luchtdruk	Vavetech	5V
8	windkracht	Vavetech	3V
9	windkracht	Samsung	5V
10	windkracht	Samsung	5V
11	windkracht	Samsung	3V
12	temperatuur	Vavetech	5V
13	temperatuur	Philips	5V
14	temperatuur	Philips	5V
15	temperatuur	Samsung	5V
16	temperatuur	Samsung	12V
17	temperatuur	Samsung	3V
18	temperatuur	Samsung	3V

inzet				
m_code	ws_id	m_id	start	eind
3681	3	2	4-10-2005	28-9-2008
3226	3	8	4-10-2005	
3847	3	13	4-10-2005	7-1-2009
3412	4	2	4-10-2005	9-9-2007
3924	5	2	4-10-2005	30-10-2008
3922	5	12	4-10-2005	
3918	1	5	14-7-2007	
3582	1	7	14-7-2007	
3637	1	12	14-7-2007	
3512	4	4	9-9-2007	20-12-2011
3134	3	6	28-9-2008	
3080	3	12	7-1-2009	
3338	5	15	1-1-2010	1-6-2010
3913	5	8	1-1-2010	
3356	2	9	27-3-2016	
3607	4	5	20-12-2011	
3721	2	1	27-3-2016	
3251	2	15	27-3-2016	
3664	5	7	14-7-2017	

FIGUUR 3

H1 SQL ZOEK EN VIND

1.1 SQL: de computer laten zoeken

Bij opdracht 2 moet je zelf zoeken in de drie tabellen van de database met gegevens van weerstations. Dat wordt al snel een vervelend klusje. Bedenk hoe langdurig jouw zoektocht naar de antwoorden zou zijn geweest, als alle weerstations van Nederland in de tabel waren opgenomen. En als we voor al die weerstations nog meer meetapparatuur zoals regenmeters, luchtvochtigheidsmeters etc. hadden toegevoegd!

Wanneer vraagstukken niet meer te overzien zijn voor mensen of je veel sneller antwoorden wilt krijgen, kun je gebruik maken van de computer. Maar hoe stel je een vraag aan de computer? Hoe leg je de computer uit naar welke gegevens je op zoek bent?
Hiervoor is een speciale taal ontwikkelt: **SQL** of *Structured Query Language*. Met SQL kun je niet alleen een database bevragen om gegevens te selecteren. Je kunt ook gegevens toevoegen, wijzigen of verwijderen en berekeningen uitvoeren, sorteren, groeperen etc. In dit hoofdstuk leer je een aantal van deze technieken om rijen uit een tabel te selecteren. Zo'n rij heet een **record**. Hieronder staan een aantal voorbeelden van sql-zoekopdrachten of **query's**.

```
SELECT * FROM stations                                  -- Toont de tabel stations
SELECT merk FROM meters                                 -- Toont de kolom merk van de tabel meters
SELECT merk,voeding FROM meters                         -- Toont zowel de kolom merk als voeding
SELECT beheerder FROM stations WHERE plaats='Groningen' -- Namen van beheerders in Groningen
```

In bovenstaande voorbeelden staan typische SQL-uitdrukkingen. Ze zijn hier voor de duidelijkheid met hoofdletters geschreven, maar dat hoeft niet perse. Met **SELECT** geef je aan dat je gegevens wilt selecteren, met **FROM** geef je aan in welke tabel of tabellen de computer moet zoeken. Achter SELECT kun je aangeven welke kolommen (informatie) je terug wilt krijgen. Met de asterisk (*) geef je aan dat je alle kolommen wilt selecteren.

Met **WHERE** kun je een voorwaarde of eis toevoegen. In de laatste query hierboven wordt een lijst met beheerders opgevraagd met de eis dat ze een weerstation in Groningen moeten beheren. Is er meer dan één voorwaarde? Gebruik **AND**: SELECT * FROM meters WHERE type='luchtdruk' AND voeding='5V'

SQL leer je het beste door het te oefenen. Van je leraar krijg je een omgeving met meerdere databases die je kunt gebruiken bij de opdrachten in deze module.

Noteer alle query's en overige antwoorden in een schrift of in een Word-document! Wanneer gevraagd wordt om iets te selecteren, noteer je de query. Wordt er een andere vraag gesteld waarvoor je een query moet gebruiken om het antwoord te vinden, dan noteer je zowel de query als het antwoord.

Opdracht 3: Oefenen met selecteren

Gebruik de database *weerstations*.

15. Voer de query's uit de theorie uit en bekijk het resultaat.
16. Selecteer alle (gegevens van) meters met een 3V-voeding.
17. Welke temperatuurmeter van het merk Samsung werkt op een 5V-voeding? Geef alleen de *m_id*.
18. Gebruik de *m_id* uit de vorige vraag om uit te zoeken hoeveel van deze meters zijn ingezet. Geef de codes (*m_code*), startdatum en einddatum van deze meters.
19. Selecteer alle (gegevens van) luchtdrukmeters van het merk Vavetech met een 5V-voeding.

Opdracht 4: Kennismaken met de database Top-2000

De top-2000 is een jaarlijks terugkerend radioprogramma. Tussen Kerst en Nieuwjaar worden op radio 2 de beste 2000 liedjes aller tijden afgespeeld. Wat die beste liedjes zijn wordt bepaald door mensen die online hun voorkeur aangeven.
Gebruik de database *Top 2000 v1*. Er is ook een database *Top 2000 v2*. Daar gaan we verderop in deze module mee aan de slag. In de database staan de *titel* van het nummer en de naam van de *artiest*. De kolom *uit* is het jaar waarin het nummer is gemaakt. De kolommen *ed2017*, *ed2018*, *ed2019*, *ed2020* en *ed2021* geven de positie van het nummer in de editie van de top 2000 van dat kalenderjaar. De waarde NULL (leeg veld) in één van deze kolommen betekent dat het nummer dat jaar geen notering in de top-2000 had.

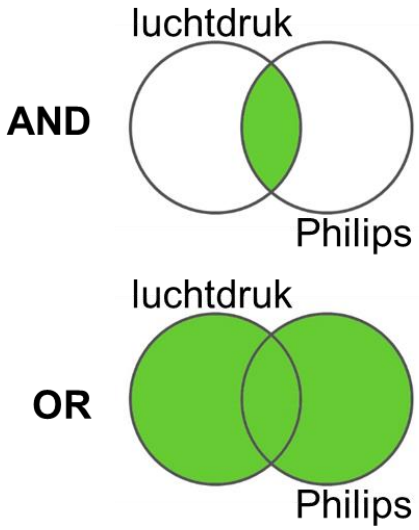
20. Voer de query *SELECT * FROM top2000* en bekijk de inhoud van de database.
21. Selecteer het record van het nummer dat op positie 1627 stond in de top-2000 van het jaar 2021?
22. In welk jaar had Maan voor het eerst een notering in de top-2000?
    Selecteer alle records met de naam van de artiest om tot je antwoord te komen.
23. Hoeveel nummers van Adele die gemaakt zijn in 2011 hebben een notering gehad in de top-2000?
    Selecteer alleen de titel en de kolom *uit*.
24. Welke artiesten hebben een notering gehad in de top-2000 met de titel *Sorry*, maar stonden niet in de editie van 2020? Zoek voor het beantwoorden van de vraag alle liedjes met de titel *Sorry*.
25. Zoek de noteringen van je favoriete artiest en of nummer op.

1.2 Logische operatoren

In paragraaf 1.1 hebben we gezien dat we twee voorwaarden kunnen combineren met de uitdrukking **AND**. Dit commando gebruikt je wanneer aan beide voorwaarden tegelijkertijd moet worden voldaan. De Venndiagrammen in figuur 4 tonen het verschil tussen **AND** en **OR**.

De linker cirkel vertegenwoordigt een verzameling meters van het type luchtdruk en de rechter cirkel vertegenwoordigt alle meters van het merk Philip. Het groen gemarkeerde deel van **AND**-tekening bevat dan alle luchtdrukmeters van het merk Philips. Bij **OR** maakt het niet uit: het groen gemarkeerde deel (alles) bevat alle meters die of van het type luchtdruk zijn of van het merk Philips.

Een bijzondere operator is **NOT** wanneer je juist wilt dat aan een eis niet wordt voldaan. En zo zijn er nog veel meer operatoren die je kunt gebruiken. De meeste zullen je bekend voorkomen van de reken- of wiskundelessen en van de programmeerlessen, toen je aan de slag ging met voorwaarden (*if*) en herhalingen (*for* en *while*).
Hieronder zie je voorbeelden van een aantal andere operatoren:



FIGUUR 4

```
SELECT * FROM meters WHERE voeding='3V' OR voeding='12V'                    -- Q1
SELECT * FROM stations WHERE NOT plaats='Groningen'                         -- Q2
SELECT * FROM inzet WHERE m_code<3200                                       -- Q3
SELECT * FROM inzet WHERE NOT eind IS null                                  -- Q4
SELECT * FROM stations WHERE ws_id>=4                                       -- Q5
SELECT * FROM meters WHERE NOT (voeding='5V' AND NOT merk='Samsung')        -- Q6 Let op de haakjes!
```

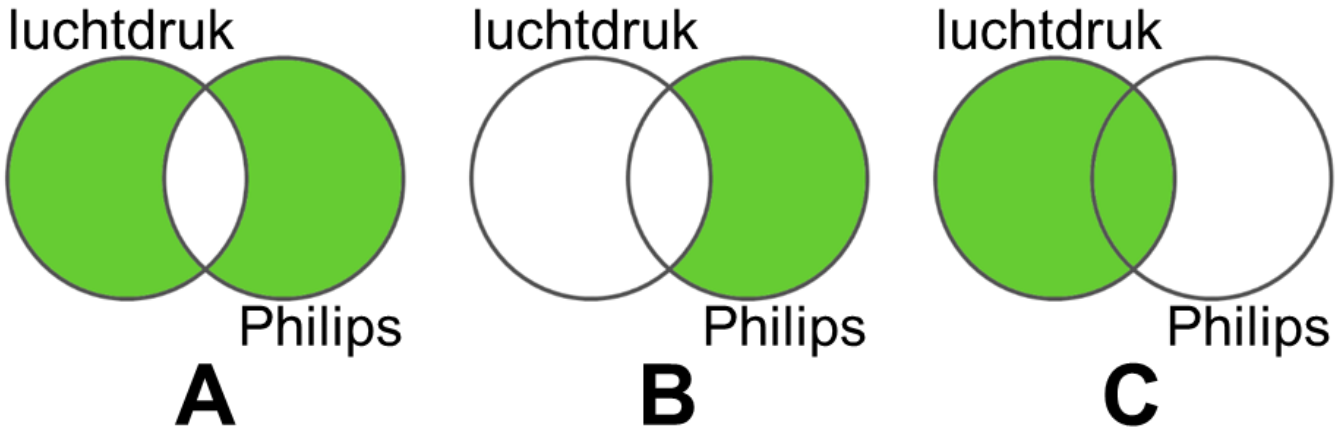
Opdracht 5: Logisch zoeken

Gebruik de database *weerstations*.

26. Lees de query's Q1 t/m Q6 uit de theorie op de vorige bladzijde en voorspel met behulp van de gegeven tabellen van de database weerstations wat het resultaat zal zijn.
27. Voer de query's nu uit en vergelijk jouw voorspelling met het resultaat. Begrijp je de query's?
28. Genereer een overzicht van alle windkrachtmeters met een een *m_id* die kleiner is dan 10.
29. Genereer een overzicht van alle temperatuurmeters die niet van Samsung zijn.

Extra opdracht 6: nog meer logisch zoeken

Gebruik de database *weerstations*.



FIGUUR 5

30. Bekijk de Venndiagrammen A, B en C in figuur 5. Beschrijf voor A, B en C in woorden welke meters er worden geselecteerd als de Venndiagrammen worden toegepast op de tabel *meters*.
31. Schrijf de query's die horen bij A, B en C.
32. Voer de query's uit. Controleer voor elke query of het resultaat overeen komt met jouw omschrijving en of het resultaat klopt met de gegeven tabel van *meters*.
33. Voorspel wat het resultaat is van de query:
    SELECT * FROM meters WHERE NOT (type='luchtdruk' AND merk='Philips')
    Controleer daarna je antwoord.
34. Voorspel wat het resultaat is van de query:
    SELECT * FROM meters WHERE NOT type='luchtdruk' AND NOT merk='Philips'
    Controleer daarna je antwoord.
35. Voorspel wat het resultaat is van de query:
    SELECT * FROM meters WHERE NOT (type='luchtdruk' OR merk='Philips')
    Controleer daarna je antwoord.

Opdracht 7: Logisch zoeken in de top-2000

Gebruik de database *Top 2000 v1*.

36. Maak een overzicht van de top-2000 van 2017 met achtereenvolgens de kolommen *ed2017*, *artiest* en *titel*. Let goed op: hoeveel records heeft jouw resultaat? (En hoeveel is logisch?)
37. Welk nummer van Oasis stond in de editie van 2017 in de top-100 van de top-2000?
38. Selecteer alle nummers van Anouk die gemaakt zijn na 2009.
39. Selecteer alle nummers van Anouk die tussen 2005 en 2010 uitkwamen (2005 en 2010 doen mee!).
40. Selecteer alle nummers uit de database van Maroon 5 en alle nummers van Stromae.
41. Welke artiesten hebben een liedje gemaakt dat voor 2000 uitkwam en de titel *One* had?
    Maak een overzicht met alleen de naam van de artiest en het jaar waarin het liedje is gemaakt.
42. (EXTRA) Hoeveel liedjes uit 1976 stonden in 2017 nog in de top-1500 van de top-2000 maar een jaar later buiten de top-1500? Genereer een overzicht van de records. (HINT 4 records is fout)

1.3 Resultaten ordenen & dubbel resultaat vermijden

We zijn inmiddels in staat om records uit een database te selecteren op basis van één of meerdere eisen. Nu willen we de records ook op volgorde kunnen leggen. Het **sorteren** van records doe je met de opdracht **ORDER BY**:

```
SELECT * FROM stations ORDER BY plaats
-- Zet de records op (alfabetische) volgorde van plaats

SELECT * FROM stations ORDER BY plaats ASC
-- Sorteer van a t/m z (hetzelfde resultaat als hierboven)

SELECT * FROM stations ORDER BY plaats DESC
-- Sorteer van z t/m a
```

Het resultaat van de query's zie je in figuur 6. De eerste twee query's geven het resultaat uit de bovenste tabel, de laatste query geeft het resultaat uit de onderste tabel.

Met **ASC** (Engels: *ascending* = oplopend) en **DESC** (Engels: *descending* = aflopend) geef je aan op welke manier je wilt sorteren.

stations

ws_id	plaats	beheerder
4	Bedum	Palsma
5	Garrelsweer	Osinga
2	Groningen	Velthuizen
3	Groningen	Grgurina
1	Zuidhorn	Van der Veen

stations

ws_id	plaats	beheerder
1	Zuidhorn	Van der Veen
2	Groningen	Velthuizen
3	Groningen	Grgurina
5	Garrelsweer	Osinga
4	Bedum	Palsma

FIGUUR 6

Stel dat je wilt weten welke bedrijven allemaal meters maken voor onze weerstations. Je zou dan met een query de kolom *merk* uit de tabel *meters* kunnen selecteren, maar dan krijg je voor elke meter het merk. Het gevolg hiervan is dat je negen keer "Samsung" op je beeldscherm krijgt. Eén keer is in dit geval wel genoeg. De oplossing is het gebruik van **DISTINCT** (Engels: uniek). De computer geeft dan alleen een nieuwe bedrijfsnaam terug als die verschilt van de reeds vermelde namen:

```
SELECT DISTINCT merk FROM meters          -- Geeft unieke merknamen

SELECT DISTINCT merk,type FROM meters     -- Geef unieke combinaties van merk en type
```

De onderste query hierboven zoekt unieke combinaties van type en merk. Hierbij maakt het niet uit of de voeding verschilt!

Opdracht 8: ORDER BY en DISTINCT

Gebruik de database *weerstations*.

43. Geef een alfabetisch overzicht van beheerders. Zorg dat alleen de naam wordt getoond.
44. Sorteer de tabel *inzet* op basis van de startdatum waarop de meter is geplaatst.
45. Voer de volgende query uit en bekijk het resultaat. Wat is het verschil met de bovenste tabel uit figuur 6? Wat moet je van dit voorbeeld leren?
    SELECT * FROM stations ORDER BY plaats,beheerder
46. Voer de volgende twee query's uit. Leg uit wat het verschil is tussen de query's:
    A: SELECT * FROM meters ORDER BY type,merk
    B: SELECT * FROM meters ORDER BY merk,type
47. Wat verandert er allemaal aan het resultaat als we query B van de vorige vraag aanpassen tot:
    SELECT merk,type,voeding FROM meters ORDER BY merk DESC,type ASC,voeding DESC
    ? Probeer het antwoord te geven zonder de query uit te voeren.
48. Kijk nog eens goed naar het resultaat van de query uit de vorige vraag. Wat valt je op aan de sortering van de kolom voeding die gesorteerd is met DESC?
49. Maak een overzicht van de verschillende meters die in de database voorkomen. Elk type meter mag maar één keer worden genoemd.
50. Wat is het resultaat van de query: SELECT DISTINCT voeding,type FROM meters ORDER BY type

Opdracht 9: ORDER BY en DISTINCT in de top-2000

Gebruik de database *top 2000 v1*.

51. Toon alle door Michael Jackson gemaakte liedjes met een notering in de top 2000, op volgorde van het jaar van uitgave. Het resultaat moet alleen de titel en het jaar bevatten, met het meest recente nummer bovenaan.
52. De artiest Sting (figuur 7; midden) was vroeger de zanger van The Police. Geef een alfabetisch overzicht van alle nummers van The Police en Sting. Vermeld alleen de artiest en de titel.
53. Sorteer het resultaat van de vorige vraag op jaar van uitgave. Lukt dit ook zonder dat het jaar van uitgave wordt getoond in het resultaat?
54. Maak een alfabetische lijst van alle artiesten die een notering in de top-20 van de top-2000 van 2018 hadden. Elke artiestennaam mag maar één keer voorkomen.
    Hoeveel verschillende artiestennamen vind je in de lijst?
55. Er staan tien liedjes van Racoon in de database. Een aantal daarvan is in hetzelfde jaar gemaakt.
    Hoeveel verschillende jaren van uitgave zijn er voor liedjes van Racoon?
    Maak een aflopend gesorteerd overzicht van deze jaren van uitgave van liedjes van Racoon. Zorg dat elk jaar maar één keer voorkomt.

FIGUUR 7 THE POLICE MET STING

Extra opdracht 10: Als je het niet precies weet

Tot nu toe hebben we query's gemaakt waarbij we steeds de exacte zoekterm (zoals naam van de artiest) wisten. Maar wat nu als je het niet helemaal weet? Daarvoor gebruiken we de operator **LIKE** in combinatie met de **wildcard**-symbolen % en _.

Gebruik de database *top 2000 v1*.

56. Bekijk het resultaat van de query:
    SELECT DISTINCT artiest FROM top2000 WHERE artiest LIKE 'Av%'
    Wat is in dit geval de betekenis van het %-teken?
57. Hoeveel unieke artiestennamen die eindigen op een i staan er in de database, denk je? Doe eerst eens een gok en maak daarna een query die het overzicht van deze namen geeft (waarbij elke naam maar één keer in de lijst mag staan).
58. Leg zorgvuldig in woorden uit wat het resultaat is van de query:
    SELECT DISTINCT artiest FROM top2000 WHERE artiest LIKE 'The%Brothers'
    Controleer daarna jouw beschrijving. Was die zorgvuldig genoeg voor alle resultaten van de query?
59. Bekijk het resultaat van de query:
    SELECT DISTINCT artiest FROM top2000 WHERE artiest LIKE 'Tom _____'
    Wat is in dit geval de betekenis van het teken _? (De query bevat na de spatie vijf opeenvolgende _'s!)
60. Hoeveel liedjes met de term Christmas in de titel staan er in de database?
61. Leg uit wat het verschil is tussen de query's:
    A: SELECT artiest,titel FROM top2000 WHERE artiest LIKE 'the%'
    B: SELECT artiest,titel FROM top2000 WHERE artiest LIKE 'the %'
62. Zie de vorige vraag. Welke resultaten volgen niet uit de ene maar wel in de andere query? Bedenk een query die als laat zien welke artiesten wel uit query A maar niet uit query B volgen.
63. Hoeveel artiestennamen van drie letters zijn er?
64. Hoeveel vermeldingen zijn er waarbij zowel de naam van de artiest als de titel de letter q bevat?
65. Bedenk zelf een gekke vraag die je buurman / - vrouw moet beantwoorden met de LIKE-operator.

FIGUUR 8 TOM WAITS

1.4 Tellen, groeperen en rekenen

Hoeveel leerlingen zitten er in havo-4? Hoeveel krentenbroden hebben we nog op voorraad? Hoeveel lampen zijn er het afgelopen jaar vervangen?

FIGUUR 9 VOORRAADBEHEER

In veel systemen met een database spelen aantallen een grote rol. Ze leveren cruciale informatie en kennis op voor bedrijven en organisaties. Vaak staan die aantallen niet als getal in de database, maar worden de aantallen berekend met SQL. Als we willen tellen, gebruiken we **COUNT** (Engels: tel of tellen).

Hieronder staan twee voorbeelden van de toepassing van COUNT:

```
SELECT COUNT(*) FROM stations WHERE NOT plaats='Groningen'
-- Tel het aantal stations dat niet in de plaats Groningen ligt.

SELECT plaats,COUNT(*) AS hoeveelheid FROM stations GROUP BY plaats ORDER BY hoeveelheid DESC
-- Tel het aantal stations per plaats en geef de kolom met dat aantal de titel hoeveelheid. Sorteer op deze hoeveelheid
```

In figuur 10 zie je het resultaat van de tweede query hierboven. De tweede kolom in deze tabel heeft de naam *hoeveelheid* meegekregen. Dit is gedaan met **AS**. Het getelde aantal is geen kolom in de oorspronkelijke database. Daarom kun je er een **ALIAS** aan meegeven: *hoeveelheid* is hier de alias.

stations

plaats	hoeveelheid
Groningen	2
Zuidhorn	1
Bedum	1
Garrelsweer	1

FIGUUR 10

De tweede query telt niet zomaar het totale aantal stations (de eerste query hierboven doet dat wel!). In plaats daarvan worden de weerstations per plaats gegroepeerd met **GROUP BY**. Het zijn dus de aantallen per plaats.

Natuurlijk kunnen we de computer niet alleen laten tellen. In de opdrachten leer je werken met bijvoorbeeld gemiddelde (**AVG** = *average*) en minimum (**MIN**).

Opdracht 11: Rekenen en groeperen

Gebruik de database *weerstations*.

66. Schrijf een query die het aantal meters met een 3V-voeding telt.
67. Genereer een tabel met de kolommen *merk* en *aantal* die het aantal verschillende types meters per merk in de tabel meters telt. Zorg dat het merk met de meeste verschillende meters bovenaan staat.
68. Voorspel wat de volgende query doet en controleer daarna het resultaat:
    SELECT MAX(m_code),start FROM inzet
69. Uit de tabel *inzet* kan worden gehaald hoe vaak een bepaald type meter (*m_id*) is ingezet. Welke meter is het vaakst ingezet of ingezet geweest?
    Maak een overzicht van *m_id* en dit aantal, op volgorde van het aantal van hoog naar laag. Deze kolom geef je de naam *N*.

Opdracht 12: Rekenen aan de top-2000 database

Gebruik de database *top 2000 v1*.

70. Maak een query die telt hoeveel liedjes van de artiest R.E.M. voorkomen in de database.
71. Wat was de hoogste notering van een R.E.M.-nummer in de Top-2000 editie 2021?
    Gebruik **MIN** om de positie van het nummer te bepalen.
72. Maak een overzicht van artiesten en het aantal liedjes dat ze in de database hebben op volgorde van het aantal (aflopend). Welke artiest heeft de meeste titels? Hoeveel?
73. (EXTRA) Voorspel wat de volgende query doet en controleer daarna het resultaat:
    SELECT COUNT(DISTINCT artiest) FROM top2000

1.5 Een query in een query

Een fan van Di-rect vraagt zich af welk nummer van hen het laagst stond in de top 2000 van 2021. Ze voert de volgende query uit:

```
SELECT titel,ed2021 FROM top2000
WHERE artiest='Di-Rect' AND NOT ed2021 IS NULL
ORDER BY ed2021 DESC
```

titel	ed2021
Wild Hearts	1775
Devil Don't Care	1443
Times Are Changing	1067
Soldier On	14

FIGUUR 11

Het resultaat van deze query zie je in figuur 11. Deze query geeft vier resultaten. Stel nu dat we alleen de titel met de laagste notering (in de top 2000 is dat een hoog getal!) als resultaat willen.

Bij vraag 71 heb je de hoogste notering voor de band R.E.M. in 2021 gevonden. Dit was wel zonder titel:

```
SELECT MIN(ed2021) FROM top2000 WHERE artiest='R.E.M.'
```

Als we de titel van het laagst genoteerde nummer van Di-rect in 2021 willen vinden, ligt deze query misschien voor de hand:

```
-- LET OP de volgende query levert een foutmelding op!
SELECT MAX(ed2021),titel FROM top2000
WHERE artiest = 'Di-Rect'
ORDER BY ed2021 DESC
```

Error: In aggregated query without GROUP BY, expression #2 of SELECT list contains nonaggregated column 'top_2000_v1.top2000.titel'; this is incompatible with sql_mode=only_full_group_by

FIGUUR 12 FOUTMELDING!

Het resultaat in figuur 12 is teleurstellend: een foutmelding! Met SQL kun je wel de laagste positie vinden (als je het gedeelte *,titel* uit bovenstaande query verwijdert), maar om dit maximum te vinden moet hij wel alle records bekijken met Di-rect als artiest. Daarbij onthoudt hij alleen het maximum, maar niet de overige gegevens van het bijbehorende record.

De oplossing is om de zoekopdracht in stapjes op te delen: eerst zoeken we de laagste positie (1775) en daarna zoeken we in de database het liedje dat hierbij hoort. Je krijgt dan een query binnen een query:

```
SELECT titel,ed2021 AS laagst FROM top2000 WHERE ed2021 =
(
    SELECT MAX(ed2021) FROM top2000 WHERE artiest = 'Di-rect'
)
```

titel	laagst
Wild Hearts	1775

FIGUUR 13 HET JUISTE RESULTAAT

Let goed op de haakjes! Tussen de haakjes (vetgedrukt) wordt eerst opgezocht dat de laagste positie de waarde 1775 heeft. Deze waarde wordt in de eerste regel gebruikt voor *ed2021* om de titel erbij te zoeken.

Opdracht 13 Dubbele top 2000-query's

74. Maak een query die als enige resultaat de titel en positie geeft van de hoogste notering van de band R.E.M. (figuur 14) in de editie van de top 2000 van 2021.
75. Voorspel wat de volgende query doet en controleer daarna het resultaat:
    SELECT titel,ed2021 AS hoogst FROM top2000 WHERE artiest='R.E.M.' AND ed2021<
    (SELECT AVG(ed2021) FROM top2000
    WHERE artiest='R.E.M.')
76. (EXTRA) Selecteer alle titels van *The Beatles* die in 2020 hoger stonden dan het hoogst genoteerde nummer van R.E.M. van dat jaar. Om welke titels gaat het?

FIGUUR 14 R.E.M.

1.6 EXTRA Het gebruik van IN en HAVING

Een query levert een verzameling van nul of meer records of andere resultaten. Het is ook mogelijk om van twee query's de uitkomsten te vergelijken, om bijvoorbeeld te zien of er overlap in resultaten is of juist niet. We leggen dit uit aan de hand van de volgende vraag:
Welke artiesten stonden in 2021 in de top-100 van de top 2000, maar een jaar eerder nog niet?
(Nota bene: dat hoeft dus niet per se met dezelfde titel te zijn!)

We maken eerst een query die een lijst maakt van alle artiesten in de top-100 van 2020:

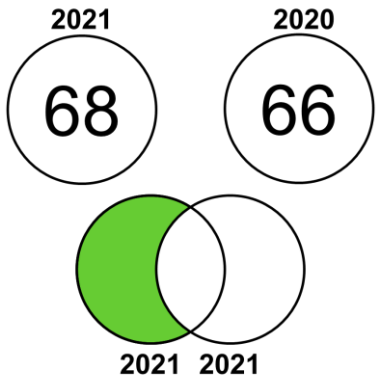
```
SELECT DISTINCT artiest FROM top2000 WHERE ed2020<=100          -- artiesten top 100 van 2020
```

Dit levert een lijst met 66 artiestennamen, omdat sommige artiesten met meerdere nummers zijn vertegenwoordigd. Van deze 66 namen moeten we uitzoeken of ze met een liedje in de top-100 van 2021 stonden. We zoeken de artiesten voor wie dit niet het geval is:

```
SELECT DISTINCT artiest FROM top2000 WHERE ed2021<=100          -- artiesten top 100 van 2021
AND artiest NOT IN                                              -- extra eis
(SELECT DISTINCT artiest FROM top2000 WHERE ed2020<=100)        -- artiesten top 100 van 2020
```

De derde regel bevat tussen haakjes de query waar we mee zijn begonnen: dit leverde 66 artiesten. De eerste regel selecteert artiesten die in 2021 een top-100-notering hadden (dat zijn 68 verschillende artiesten).

De tweede regel stelt de eis dat van deze tweede groep artiesten er niemand ook in de lijst artiesten van 2020 mag zitten. Figuur 15 markeert nu de artiesten waar we naar op zoek zijn. Het groene deel bevat de artiesten die wel in 2021 wel maar in 2020 niet in de top-100 stonden. De vraag is:
Wie van die 68 artiesten zit niet in de lijst met 66 artiesten?



FIGUUR 15

We gebruiken hiervoor **NOT IN**. Wil je juist op zoek naar artiesten die in beide resultaten terug komen, dan gebruik je **IN**.

Wanneer we een berekening uitvoeren met bijvoorbeeld **COUNT**, **AVG,MAX** of **MIN** en we willen dat de uitkomst een bepaalde eigenschap heeft, dan gebruiken we **HAVING**. Ook hier een voorbeeldvraag:
Welke artiesten staan met 20 titels of meer in de database?

Bij opdracht 12 hebben we een overzicht van het aantal titels per artiest gemaakt met de query:

```
SELECT artiest,COUNT(*) AS aantal FROM top2000 GROUP BY artiest ORDER BY aantal DESC
```

Hiermee krijgen we ook artiesten die maar met één of twee liedjes in de database staan. Met de toevoeging **HAVING** en de juiste eis kunnen we de voorbeeldvraag beantwoorden:

```
SELECT artiest,COUNT(*) AS aantal FROM top2000 GROUP BY artiest
HAVING aantal >= 20 ORDER BY aantal DESC
```

Kijk goed naar het vetgedrukte deel: de **alias** *aantal* die in de eerste regel wordt gemaakt, wordt in de tweede regel opnieuw gebruikt.

EXTRA opdracht 14: Top 2000 met IN en HAVING

77. Voer de query met de **NOT IN**-constructie uit de theorie uit. Hoeveel resultaten krijg je?
78. Harald zegt dat de query uit de vorige vraag hetzelfde resultaat oplevert als de query:
    SELECT DISTINCT artiest FROM top2000 WHERE ed2021<=100
    AND NOT ed2020<=100
    Vind jij dat ook? Zo niet, leg dan uit wat het verschil is in resultaat tussen de twee query's.
79. In de **NOT** IN-query van vraag 77 staat *artiest*. Vervang dit door *titel*.
    Verwacht je nu evenveel resultaten als bij de vorige vraag? Of meer of minder?
80. Genereer een overzicht van alle liedjes in de database van artiesten die in 2018 een top-10-notering hadden. Zorg voor een tabel met artiest, titel en het jaar (waarin het liedje gemaakt is: *uit*). Sorteer de tabel op naam van de artiest en zorg dat binnen die sortering de liedjes op volgorde van jaar staan.
81. Voer de query met de **HAVING**-constructie uit de theorie uit. Hoeveel resultaten krijg je?
82. Maak een overzicht van artiesten met meer dan drie titels in de top-100 van de top 2000 van 2018. Toon de naam van de artiest en het aantal titels, gesorteerd van hoog naar laag. Zijn er meer artiesten met hetzelfde aantal? Zorg dan dat ze in alfabetische volgorde worden getoond.
83. Hoeveel titels komen vaker dan één keer voor, denk je? Doe eerst een schatting en maak daarna een query die een overzicht geeft van de titel en het aantal keren dat hij in de database voorkomt. Uiteraard zet je de titel die het vaakst voorkomt bovenaan!

Extra opdracht 15: HAVING en ALL

Bij de vorige opdracht zochten we naar titels van liedjes waar meerdere artiesten mee in de top-2000 staan. Hoe selecteer je nu alleen die titel die het meest voorkomt?

Gebruik de database *top 2000 v1*.

84. Bekijk het resultaat van de query:
    SELECT titel FROM top2000 GROUP BY titel
    HAVING COUNT(titel) >= ALL
    (SELECT COUNT(titel) from top2000 GROUP BY titel)
    Wat is de betekenis van het vetgedrukte deel van bovenstaande query? Wat is het resultaat?
85. Maak een overzicht van de artiesten die een liedje hebben met de titel die het vaakst voorkomt (het resultaat van de vorige vraag) op volgorde van het jaar van uitgave.
    LET OP: je mag de titel niet in je query invullen. Het is de bedoeling dat je de query van de vorige vraag uitbreidt om tot je antwoord te komen.

FIGUUR 16

Extra opdracht 16: Rekenen met datums

Gebruik de database *weerstations*.

86. Voer de volgende query uit en leg aan de hand van de uitkomst uit wat de functie **DATEDIFF** doet:
    SELECT m_code, DATEDIFF(eind,start) as duur,eind,start FROM inzet WHERE eind>0
    ORDER BY duur DESC
87. Wat is de betekenis van het resultaat dat je nu hebt gekregen? En wat is daarbij de eenheid van de uitkomst van **DATEDIFF**?
88. Maak een overzicht van alle meters die meer dan duizend dagen hebben gewerkt, voordat ze werden vervangen.
89. Gebruik https://www.w3schools.com/sql/func_mysql_datediff.asp om een overzicht te genereren van meters die nog niet zijn vervangen en minder dan 500 dagen (sinds vandaag) in gebruik zijn, op volgorde van het aantal dagen (oplopend).