BIOINFORMATICA

INFORMATIESYSTEMEN MET PYTHON_

SUBQUERY'S EN GROEPFUNCTIES

HAN_UNIVERSITY OF APPLIED SCIENCES

STUDIEWIJZER

Les	Onderwerp	Theorie
1	Introductie tot Informatie Systemen	H1 Introduction to Database Development H2 Entity Relationships H3 Complex Relationships H4 Logical Database Design
2	Database ontwerp Constraints en normaliseren	H5 Normalisation H6 Introduction to Oracle SQL H7 Foreign Keys
3	SQL, operatoren en functies	H8 Selecting data from a Table
4	Rijen ophalen uit meerdere tabellen	H9 Selecting data from multiple tables
5	Subquery's en groepfuncties	H10 Subqueries and Group Functions
6	SQL Embedden in Python Set operatoren	
7	Python en SQL: mogelijkheden en risico's	

AGENDA

Subquery's

Subquery's met groepsfuncties

Gecorreleerde subquery’s

Groepen met groepsfuncties

Having by clausule

SUBQUERY

Welke studenten zitten in de klas met iemand die in Nijmegen woont?

SUBQUERY

Een subquery is een query in een query

Subquery's maken het mogelijk om tijdens het uitvoeren van de hoofdquery een extra query uit te voeren

Ook wel inner - en outer query

SUBQUERY

Stel je wilt weten wie zit in de klas met iemand die in Nijmegen woont?

Je zult twee vragen stellen

1. Wie woont er in Nijmegen?
2. Wie zit bij die student in de klas?

1) WIE WOONT ER IN NIJMEGEN?

```
select *
from   student
where  woonplaats = ‘Nijmegen’;
```

voornaam	tussenvoegsels	achternaam	woonplaats	klas_id
Ernst	<null>	Kuipers	Nijmegen	3

OPHALEN VAN KLAS_ID

Van de rij gegevens wil je alleen het klas_id weten

In dit geval dus klas_id = 3

2)WIE ZIT ER BIJ … IN DE KLAS?

```
select *
from   studenten
where  klas_id = 3
```

voornaam	tussenvoegsels	achternaam	woonplaats	klas_id
Sigrid	<null>	Kaag	Amsterdam	3
Ernst	<null>	Kuipers	Nijmegen	3
Wopke	<null>	Hoekstra	Millingen	3
Eric	van der	Burg	Elst	3
Rob	<null>	Jetten	Utrecht	3
Alexandra	van	Huffelen	Groesbeek	3
Hans	<null>	Vijlbrief	<null>	3

EEN SUBQUERY GEBRUIKEN

Een subquery is een query in een query

Hierdoor kun je in een keer de vraag beantwoorden

2)WIE ZIT ER BIJ … IN DE KLAS?

```
select *
from   studenten
where  klas_id = (select klas_id
                  from studenten
                  where woonplaats='Nijmegen')
```

voornaam	tussenvoegsels	achternaam	woonplaats	klas_id
Sigrid	<null>	Kaag	Amsterdam	3
Ernst	<null>	Kuipers	Nijmegen	3
Wopke	<null>	Hoekstra	Millingen	3
Eric	van der	Burg	Elst	3
Rob	<null>	Jetten	Utrecht	3
Alexandra	van	Huffelen	Groesbeek	3
Hans	<null>	Vijlbrief	<null>	3

WAT IS EEN SUBQUERY?

Een subquery is een query met dezelfde opbouw als een query

```
( SELECT [DISTINCT]
   subquery_select_list
  FROM {table_name}
  [WHERE search_conditions]
  [GROUP BY aggregate_expression [,
   aggregate_expression] ...]
  [HAVING search_conditions] )
```

SUBQUERY

Een subquery is een gewone query met alle clausules

Een clausule is echter niet mogelijk in een subquery, welke zal dat zijn?

Order by

Subquery kan resultaat niet manipuleren

Order by zal resultaat hoofdquery sorteren

REGELS

Een subquery staat rechts van de **operator** in de where clausule

De resultante van de subquery moet **compatibel** zijn met de expressie links van de subquery

REGELS

Wie zit er bij iemand in de klas die in Utrecht woont?

SUBQUERY'S EN OPERATOREN

Als uit een subquery meerdere resultaten kunnen voortvloeien dan moet de operator dat ondersteunen

Gebruik in dat soort gevallen dus een *in* operator

Net zoals bij gewone query's

```
select *
from   studenten
where  klas_id in (1, 1, 2)
```

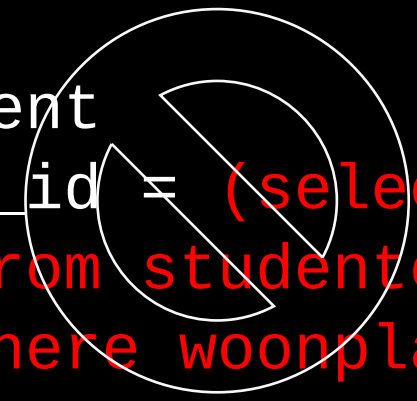
WIE ZITTEN ER BIJ IEMAND UIT UTRECHT IN DE KLAS?

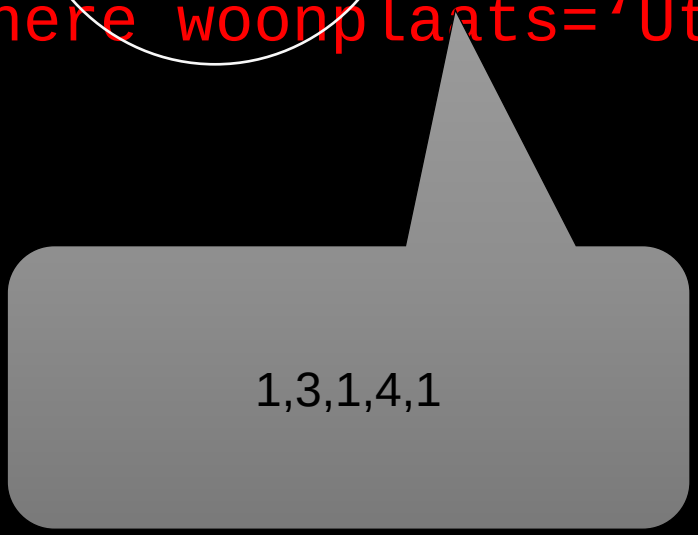
```
select *
from student
where klas_id = (select klas_id
                 from student
                 where woonplaats = ‘Utrecht')
```

[21000][1242] Subquery returns more than 1 row

MEERDERE WAARDES SUBQUERY

```
select *
from   student
where  klas_id = (select klas_id
                  from studenten
                  where woonplaats='Utrecht')
```





MEERDERE WAARDES SUBQUERY

```
select *
from   student
where  klas_id in (1,3,1,4,1)
```

De in operator is nodig omdat meerder waardes terug **kunnen** komen uit de subquery

NESTING

Subquery's mogen zelf ook weer subquery's bevatten die ook weer subquery's mogen bevatten enzovoorts

Dit principe noemen we **nesting**, net zoals dat bij loops en if statements nesting heet

AGENDA

Subquery's

Subquery's met groepsfuncties

Gecorreleerde subquery’s

Groepen met groepsfuncties

Having by clausule

GROEPSFUNCTIES

In SQL kunnen we groepsfuncties gebruiken

Groepsfuncties zijn functies die over een hele groep een uitspraak doen

GROEPSFUNCTIES

Naam	
sum()	sommering van waardes
count()	tellen van aantal resultaatrijen
min()	minimale waarde in groep
max()	maximale waarde in groep
avg()	gemiddelde waarde van een groep getallen

WAT IS HET GEMIDDELDE STUDENTENNUMMER?

```
select avg(student_nr)
from   student
```

VOORBEELD GROEPSFUNCTIE

Bepaal wat de geboortedatum is van de jongste student

```
select max(geb_datum)
from   student
```

1997-12-13

WIE HOORT BIJ DEZE DATUM?

```
select max(geb_datum)
from   student
```

1999-01-31

```
select *
from   student
where  geb_datum = '1999-01-31'
```

	student_nr	voornaam	tussenvoegsels	achternaam	geb_datum
1	12	Kim	van	Pleggen	1999-01-31

IN EEN QUERY

```
select max(geb_datum)
from   student
```

1999-01-31

```
select *
from   student
where  geb_datum =  (select max(geb_datum)
                     from   student)
```

	student_nr	voornaam	tussenvoegsels	achternaam	geb_datum
1	12	Kim	van	Pleggen	1999-01-31

GROEPSFUNCTIES

Groepsfuncties als **sum, count, max** en **min** retourneren altijd slechts 1 waarde

Wanneer je deze functies gebruikt kun je dus de = operator gebruiken

GROEPSFUNCTIES

De groepsfuncties **avg** en **sum** zijn alleen van toepassing op getallen

count, **max** en **min** kun je ook toepassen op **datums** en **teksten**

GROEPSFUNCTIE COUNT

Functie	Resultaat
count(*)	Telt het aantal resultaatrijen
count(geb_datum)	Telt het aantal geboortedatums die niet null zijn
count (distinct woonplaats)	Telt het aantal verschillende woonplaatsen

GROEPSFUNCTIES

Groepsfuncties kun je opnemen **in** de **select** clausule maar **niet in** de **where** clausule

Als je een vergelijking wilt maken in de where clausule met een groepsfunctie dan moet je die opnemen in een subquery

QUERY

Schrijf de query die de studenten toont met een studentnummer lager dan het gemiddelde studentnummer

AGENDA

Subquery's

Subquery's met groepsfuncties

Gecorreleerde subquery’s

Groepen met groepsfuncties

Having by clausule

GECORRELEERDE SUBQUERY

Een subquery met een relatie naar de hoofdquery noemen we een gecorreleerde subquery

In de vraagstelling is meestal een verwijzing naar een groep waar een rij deel van uitmaakt

VRAAG

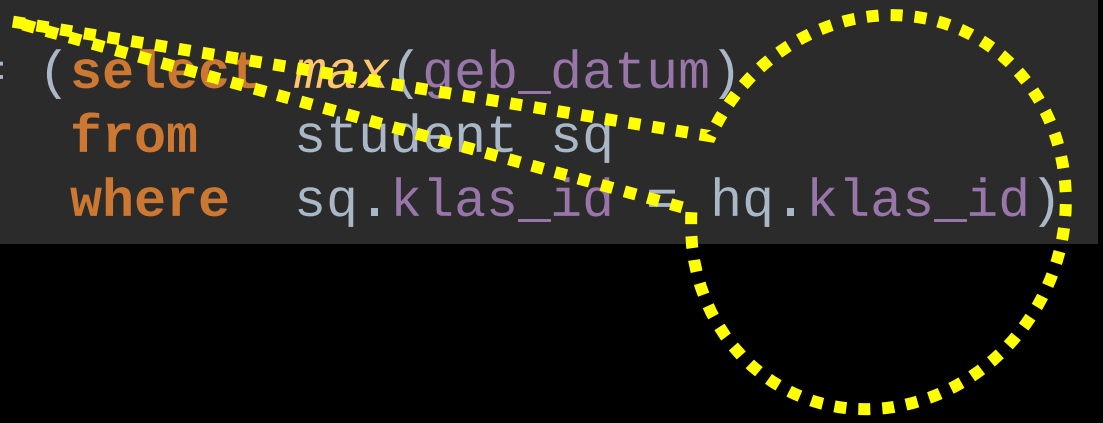
Welke student is van zijn of haar klas de jongste student?

Zijn of haar verwijst naar de klas, dus per student (rij) kijk je naar zijn of haar hele klas

QUERY

Welke student is van zijn of haar klas de jongste student?

```
select voornaam, klas_id
from   student hq
where  geb_datum = (select max(geb_datum)
                    from   student sq
                    where  sq.klas_id = hq.klas_id);
```



AGENDA

Subquery's

Subquery's met groepsfuncties

Gecorreleerde subquery’s

Groepen met groepsfuncties

Having by clausule

GROUP BY

De group by clausule maakt het mogelijk om rijen in te delen in groepen

HET SELECT STATEMENT MET GROUP BY

```
select <kolomnamen|*>
from   <tabel>
group by <kolomnaam>
```

Group by clausule maakt een indeling in groepen

GROUP BY

Wat is het aantal studenten per klas?

```
select count(*)
from   studenten
group by klas_id
```

GROUP BY

Wat is het aantal studenten per klas?

```
select klas_id, count(*)
from   studenten
group by klas_id
```

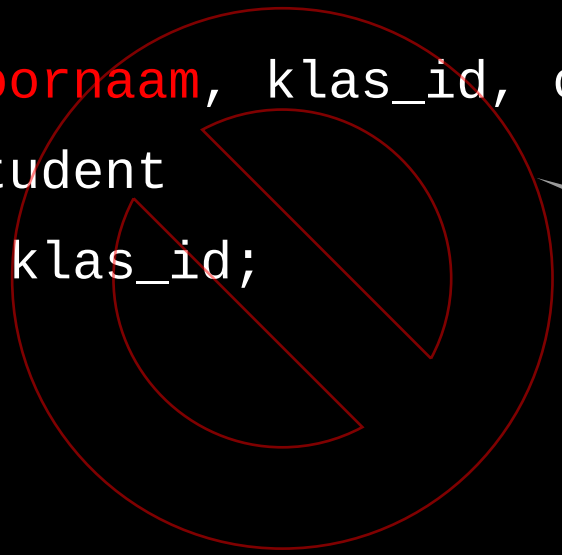
GROUP BY

Wat is het aantal studenten per klas?

Maar met de studentnamen…

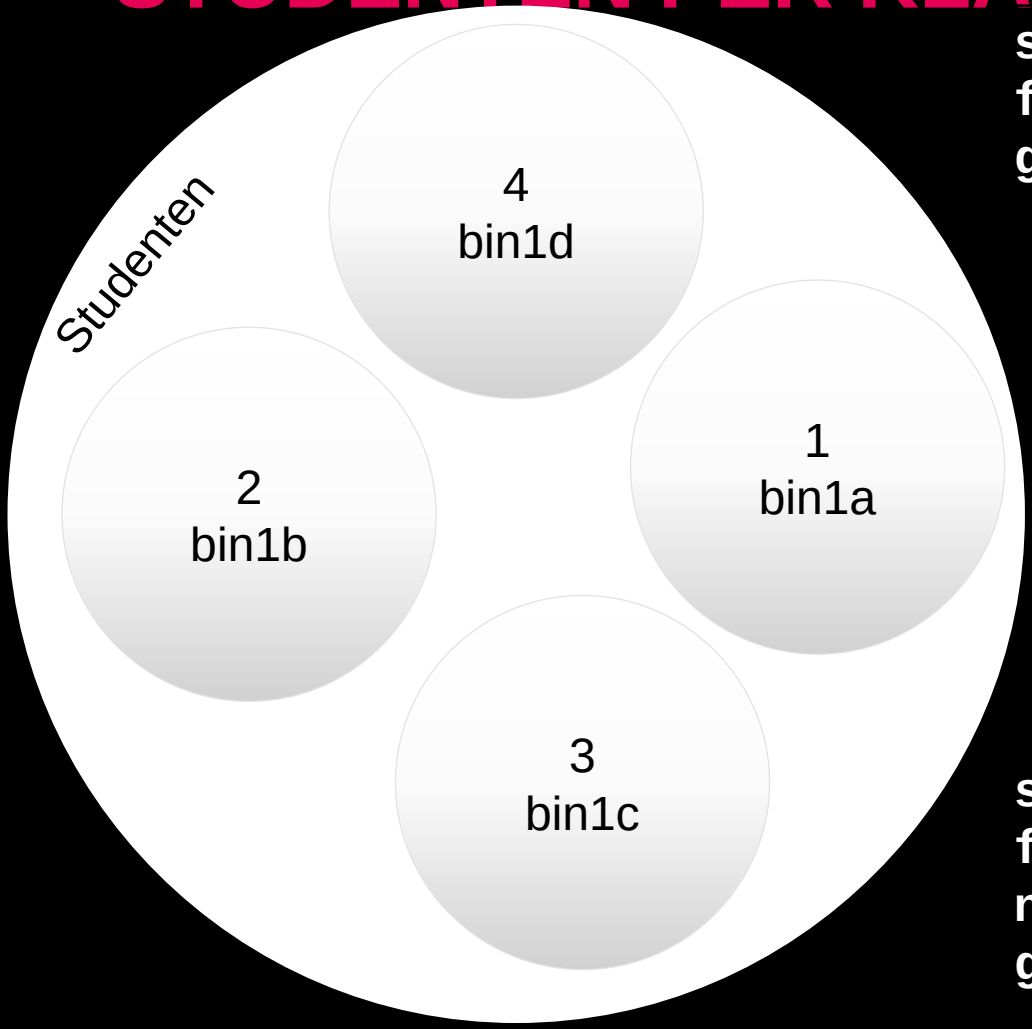
Wat gaat hier fout?

```
select voornaam, klas_id, count(*)
from   student
group by klas_id;
```



Voornaam is geen eigenschap van de groep

STUDENTEN PER KLAS_ID




```
select klas_id, count(*)
from   student
group by klas_id
```

```
select klas_id, max(geb_datum)
from   student
group by klas_id
```

```
select avg(studnr), count(*)
from   student
group by klas_id
```

```
select klas_naam, count(*)
from   student
natural join klas
group by klas_naam
```

GROUP BY

Alles wat in de group by clausule staat om te groeperen mag je opnemen in de select clausule

Het is een eigenschap van de subgroep

AGENDA

Subquery's

Subquery's met groepsfuncties

Gecorreleerde subquery’s

Groepen met groepsfuncties

Having by clausule

HAVING CLAUSULE

Een having clausule is altijd in combinatie met een group by clausule

De having clausule kan filteren op eigenschappen van groepen

HAVING CLAUSULE

De **having** clausule kan in tegenstelling tot de **where** clausule **wel groepsfuncties** bevatten!

HAVING

De having clausule kan in combinatie met group by een filter leggen op groepen

In de having clausule kunnen ook groepsfuncties worden opgenomen

VRAAG

Schrijf de query om het aantal genen per biotype te bepalen

Toon alleen de biotypes waar meer dan 200 genen voor zijn gevonden

HAVING

```
select biotype
,      count(*)
from   gene
group by biotype
having count(*) > 200
order by 2 desc
```

VRAAG

Schrijf de query om het aantal genen per biotype te tellen

Toon alleen de genen van het biotype 'protein_coding'

HAVING – WAT KAN BETER?

```
select biotype
,      count(*)
from   gene
group by biotype
having biotype = 'protein_coding'
order by 2 desc
```

De query werkt en geeft het juiste resultaat. Maar kan het beter?

VRAAG

Hoe toon je per biotype het gemiddelde GC percentage?

Toon alleen die biotypes waarvan er meer dan 1000 genen aanwezig zijn.

QUERY

```
select biotype
,      floor(avg(value))
from   gene
  natural join gene_attrib
  join attrib_type using (attrib_type_id)
where code = 'GeneGC'
group by biotype
having count(*) > 1000
```

HET SELECT STATEMENT MET HAVING

```
select <kolomnamen|*>
from   <tabel>
group by <kolomnaam>
having <conditie>
```

De having clausule kan alleen in combinatie met group by worden opgenomen

SAMENVATTING

Analytische functies

- avg
- min
- max
- count
- sum

Group by geeft de mogelijkheid uitspraken te doen over een groep

Having geeft de mogelijkheid te filteren op groepseigenschappen

SAMENVATTING

```
select <kolomnamen|*>
from   <tabel>
where  <conditie>
group by <kolomnaam>
having <conditie>
order by <kolom>
```

OPDRACHT

- Schrijf een query om iedereen te tonen die in de klas zit bij iemand uit Apeldoorn

- In welke woonplaatsen is de jongste student geboren voor 1975?

APPENDIX: GOOGLE BIG QUERY

BIGQUERY

Google Big Query is een SQL interface tot Google Cloud Data warehouse en onderdeel van Google Cloud Platform
(Google Cloud Console)

Dit betreft de opslag en het ophalen van data in de grootte van Terabytes

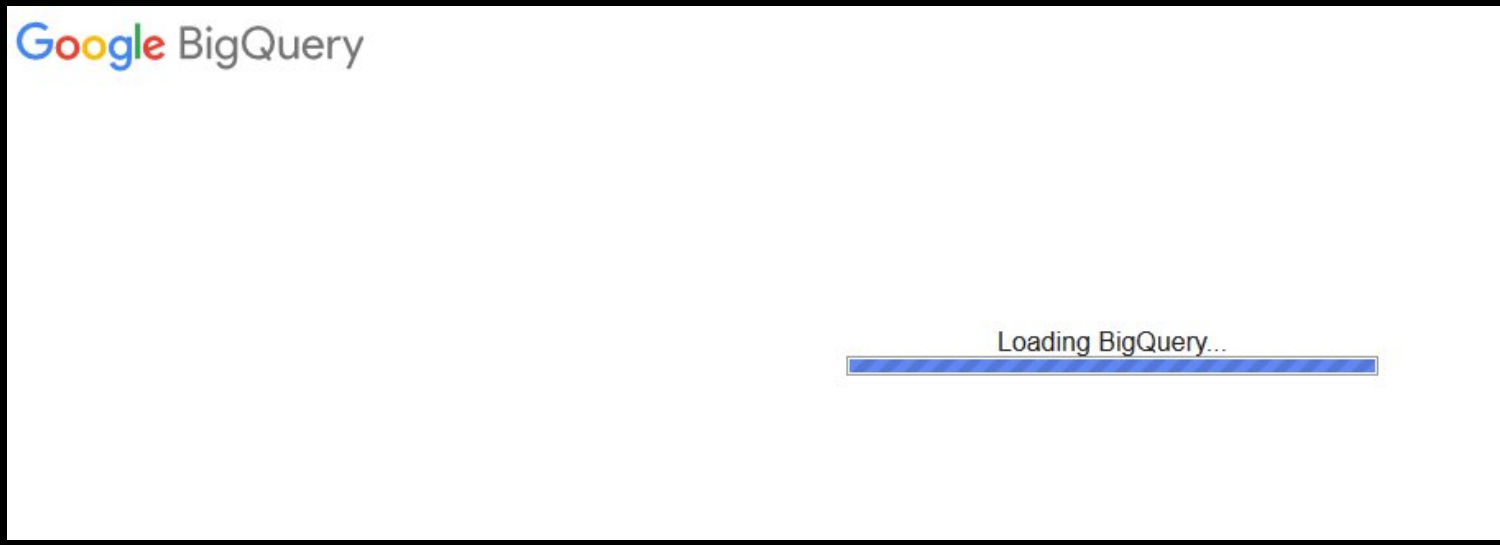
DEMO/OPDRACHT

- Toon pathogene varianten op chromosoom 12

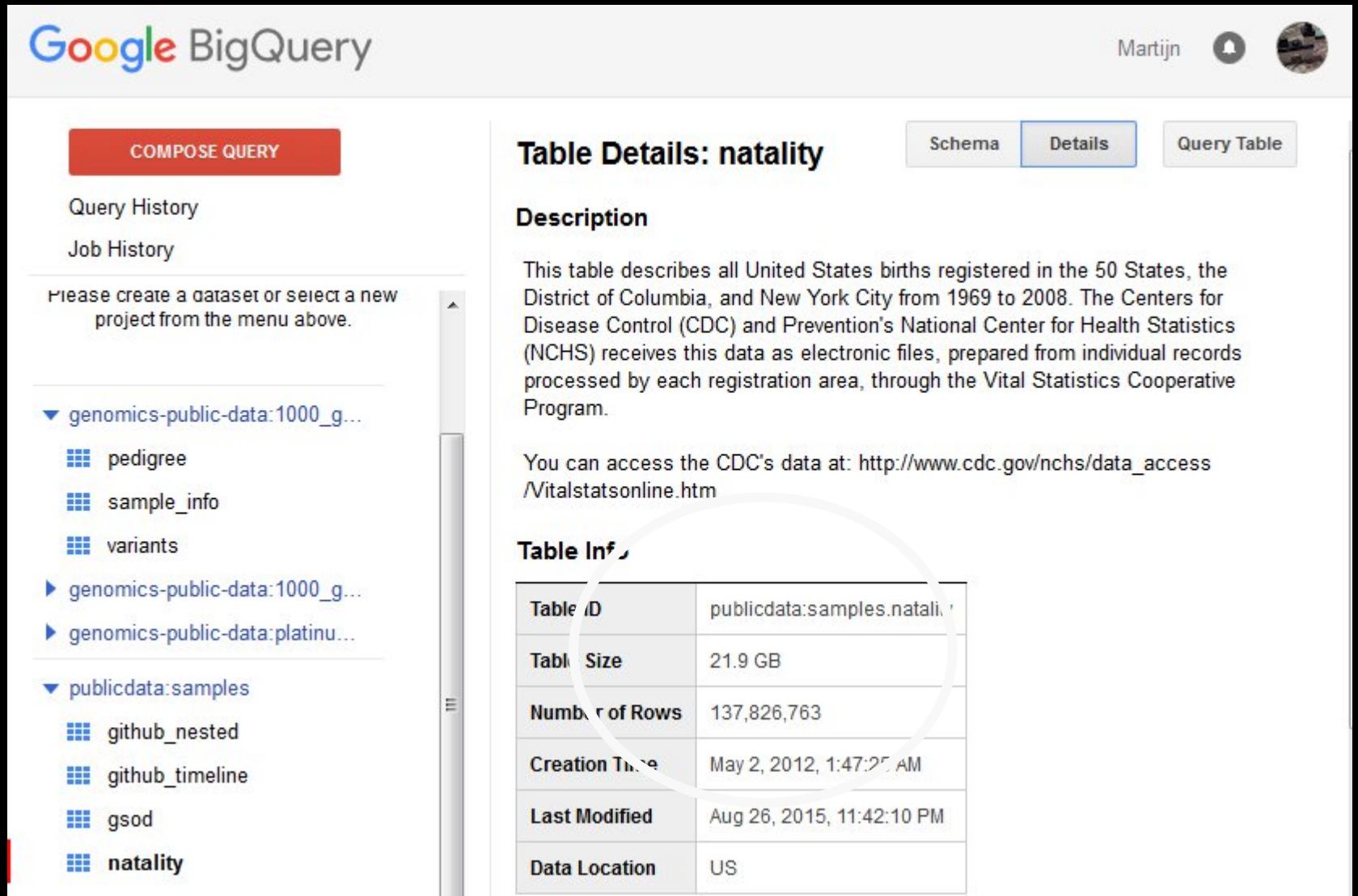
  tip: human_variant_annotation

- Toon het aantal pathogene varianten per chromosoom

GOOGLE BIG QUERY



NATALITY


Google BigQuery
Martijn
COMPOSE QUERY
Query History
Job History
Please create a dataset or select a new project from the menu above.
genomics-public-data:1000_g...
pedigree
sample_info
variants
genomics-public-data:1000_g...
genomics-public-data:platinu...
publicdata:samples
github_nested
github_timeline
gsod
natality
Table Details: natality
Schema
Details
Query Table
Description
This table describes all United States births registered in the 50 States, the District of Columbia, and New York City from 1969 to 2008. The Centers for Disease Control (CDC) and Prevention's National Center for Health Statistics (NCHS) receives this data as electronic files, prepared from individual records processed by each registration area, through the Vital Statistics Cooperative Program.
You can access the CDC's data at: http://www.cdc.gov/nchs/data_access/Vitalstatsonline.htm
Table Info
Table ID
publicdata:samples.natali
Table Size
21.9 GB
Number of Rows
137,826,763
Creation Time
May 2, 2012, 1:47:25 AM
Last Modified
Aug 26, 2015, 11:42:10 PM
Data Location
US

Tel het aantal rijen in de tabel natality

Wat is het maximum aantal sigaretten per dag dat is gerookt tijdens de zwangerschap?

Hoeveel jongens vs. meisjes staan er in de tabel?

Wat is het gemiddelde gewicht van jongens bij geboorte?

Wat is het gemiddelde gewicht van jongens bij geboorte als de moeder heeft gerookt tijdens de zwangerschap?

BIG QUERY

Kun je met een query aantonen dat alcoholgebruik van invloed is op het geboortegewicht?

Kun je met een query aantonen dat het gebruik van sigaretten en/of alcohol van invloed is op de apgar score?

GEBOORTEGEWICHT PER WEEKDAG (MEISJES)

New Query

```
1  SELECT
2    avg(weight_pounds)
3  FROM
4    [publicdata:samples.natality]
5  WHERE
6    is_male = FALSE
7  GROUP BY
8    wday
```

Query Results Oct 18, 2015, 10:23:05 PM

Table | JSON

Row	f0_
1	7.0967779405454126
2	7.105488662122104
3	7.186678839149218
4	7.197980914333184
5	7.166436097235965

ALCOHOL USAGE

New Query

```
SELECT
  avg(weight_pounds)
FROM
  [publicdata:samples.natality]
WHERE
  alcohol_use = false
GROUP BY
  wday
```

RUN QUERY | Save Query | Save View

Query Results Oct 18, 2015, 10:27:44 PM

Table | JSON

Row	f0_
1	7.324945821979766
2	7.34647671682341
3	7.313721964450922
4	7.222452965698626
5	7.330721005826806

New Query

```
SELECT
  avg(weight_pounds)
FROM
  [publicdata:samples.natality]
WHERE
  alcohol_use = true
GROUP BY
  wday
```

RUN QUERY | Save Query | Save View

Query Results Oct 18, 2015, 10:27:32 PM

Table | JSON

Row	f0_
1	6.907193636448581
2	6.930765251771114
3	6.793723191274106
4	6.907984347645765
5	6.915682964579404

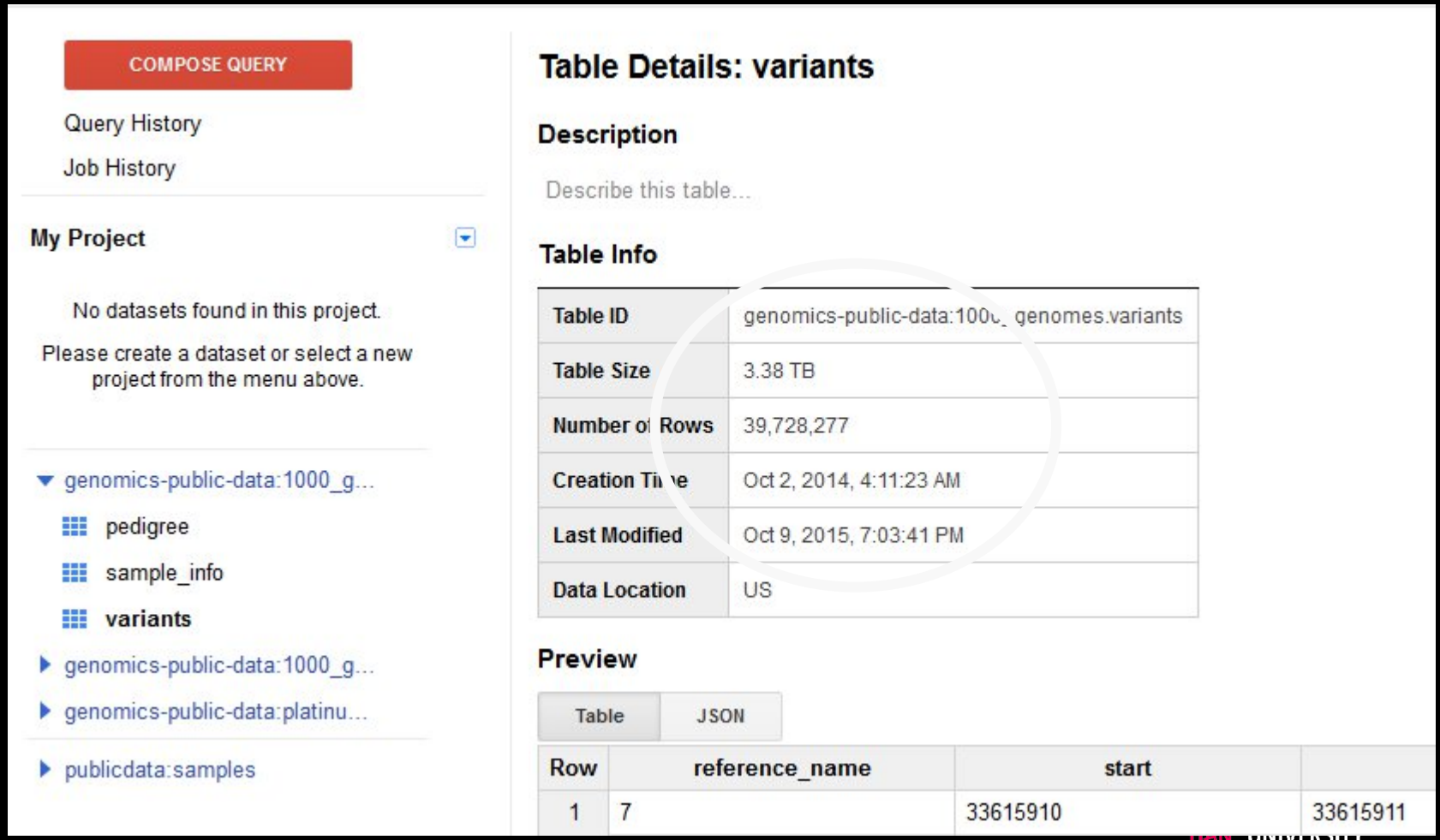


Table Details: variants

Description

Describe this table...

Table Info

Table ID	genomics-public-data:1000_genomes_phase_3.variants
Table Size	9.82 TB
Number of Rows	79,479,278

DIVERSE TABELLEN IN BIGTABLE

VERANTWOORDING

In deze uitgave is géén auteursrechtelijk beschermd werk opgenomen

Alle teksten © Martijn van der Bruggen/HAN tenzij expliciet externe bronnen zijn aangegeven

Screenshots op basis van eigen werk auteur en/of vernoemde sites

Eventuele images zijn opgenomen met vermelding van bron

SAMENVATTING

Prachtige en verschillende creatieve slimme oplossingen

Er zijn meerdere goede oplossingen mogelijk

Net als bij Python is het nodig om een goede probleemanalyse te maken

Compacte code kost veel tijd