

BIOINFORMATICA

INFORMATIESYSTE MEN MET PYTHON_

DATABASE ONTWERP EN SQL

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 tabellen	H9 Selecting data from multiple tables
5	Subqueries en groepfuncties	H10 Subqueries and Group Functions
6	SQL Embedden in Python	
7	Python en SQL: mogelijkheden en risico's	

AGENDA

- Het Fysieke Model
- SQL IDE's
- DDL: creatie tabellen
 - create, alter, drop
- DML: wijzigen inhoud tabellen
 - **insert, delete, update**
- DRL: ophalen gegevens uit tabellen
 - select

STAPPEN IN ONTWIKKELING DATABASE



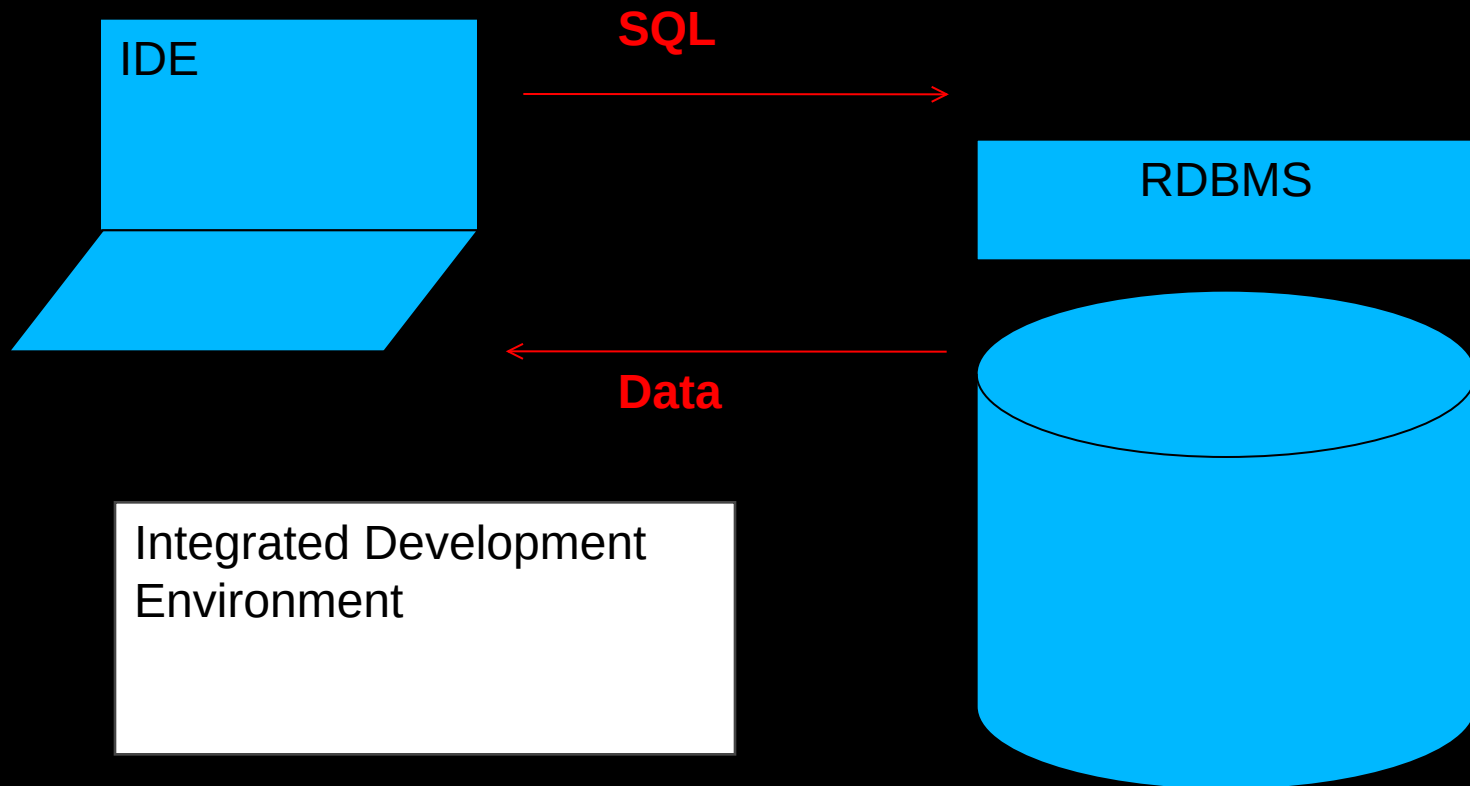
FYSIEKE MODEL

- Het fysieke datamodel is geschikt om de database te beschrijven

AGENDA

- Het Relationele Model
- SQL IDE's
- DDL: creatie tabellen
 - create, alter, drop
- DML: wijzigen inhoud tabellen
 - **insert, delete, update**
- DRL: ophalen gegevens uit tabellen
 - select

WERKEN MET EEN RELATIONELE DATABASE



Integrated Development
Environment

DATAGRIP

- Met **DataGrip** is het mogelijk de database te benaderen
- Gebruik daarvoor de gegevens zoals deze onder het kopje **Praktijk** van **Onderwijsonline** staan

AGENDA

- Het Relationele Model
- SQL IDE's
- DDL: creatie tabellen
 - create, alter, drop
- DML: wijzigen inhoud tabellen
 - **insert, delete, update**
- DRL: ophalen gegevens uit tabellen
 - select

VRAAG:

- Wat kun je allemaal met de “vraagtaal” SQL?

DE 5 TAALGROEPEN

Groep	Doel	Statements
DRL	Data Retrieval	select
DML	Data Manipulation	update delete insert
DDL	Data Definition	create alter drop
TCL	Transaction Control	commit rollback
ACL	Access Control	grant revoke

CREATE

```
CREATE TABLE student (  
    stud_id          CHAR(9),  
    voornaam         VARCHAR(25),  
    tussenv          VARCHAR(10),  
    achternaam       VARCHAR(25),  
    geb_datum        DATE,  
    reistijd          DECIMAL(7,2)  
);
```

stud_id	voornaam	tussenv	achternaam	geb_datum	reistijd

ALTER

- Met alter kun je de structuur van een tabel wijzigen
- Dit gebruik je niet vaak omdat dit een ***dure*** database actie is

DROP

- `drop table student;`
- Het droppen van een tabel is (afhankelijk van de database) niet meer ongedaan te maken.

stud_id	voornaam	tussenv	achternaam	geb_datum	reistijd

NULL

- Null (spreek uit nil) is in database jargon leeg
- Null is niet van toepassing of onbekend
- Null is iets anders dan 0 of een spatie!

Null: niet van
toepassing

Null: onbekend

stud_id	Voornaam	tussenv	achternaam	geb_datum
546271	Henk		Potter	
552063	Gonny		Henkes	10-oct-1999
553958	Rick	van	Kessel	30-feb-1989

DATATYPES

Type	Betekenis
varchar (x)	Tekst van lengte x
char (x)	Tekst van lengte x
int	Geheel getal
decimal (x,y)	Getal met x posities, met een <i>decimal seperator</i> op positie y
date	Datum
CLOB	Character Large Object: grote teksten gigabytes
BLOB	Binary Large Object: grote binaire bestanden van gigabytes

CONSTRAINTS

- Er zijn 5 constraints die we aan tabellen mee kunnen geven
- Constraints zijn beperkingen/aanwijzingen die gelden voor een kolom of groep van kolommen

DE CONSTRAINTS

	Naam	Afk	Betekenis
1	Primary Key	PK	Identificeert een rij
2	Foreign Key	FK	Verwijst naar een andere rij
3	Unique	UK	Uniek in die kolom
4	Not null	NN	Mag niet leeg zijn
5	Check	C	Controle op inhoud

CONSTRAINT 1: PRIMARY KEY

- De primary key is de identificerende sleutel
- Is nooit leeg (nooit null)
- Bevat een unieke waarde
- Maximaal een PK per tabel

CONSTRAINT 1: PRIMARY KEY

```
create table student
(
  student_nr      int      primary key,
  voornaam        varchar(11) not null,
  tussenvoegsels  varchar(10) null,
  achternaam      varchar(20) not null,
  geb_datum       date      null,
  woonplaats      varchar(20) null,
  email           varchar(25) null,
  mobiel          varchar(15) null,
  klas_id         int       null,
  slb_id          int       null,
  constraint AK_0
    unique (email)
);
```

CONSTRAINT 2: FOREIGN KEY

- Is de refererende sleutel
- Verwijst naar een andere kolom
- Kan leeg (null) zijn
- Is niet uniek

student

stud_id	voornaam	klas_id
496810	Gonny	4
513767	Henk	4
481236	Margo	1
508953	Martijn	

klas

klas_id	naam	aantal
1	Bi1a	13
2	Bi1b	13
3	Bi1c	14
4	Bi1d	12

FOREIGN KEY

```
create table klas
```

```
(  
  klas_id    int    primary key,  
  klas_naam  varchar(20) null  
);
```

```
create table student
```

```
(  
  student_nr    int    not null  
    primary key,  
  voornaam      varchar(11) not null,  
  tussenvoegsels varchar(10) null,  
  achternaam    varchar(20) not null,  
  geb_datum     date    null,  
  woonplaats    varchar(20) null,  
  email         varchar(25) null,  
  mobiel        varchar(15) null,  
  klas_id       int    null,  
  slb_id        int    null,  
  constraint AK_0  
    unique (email)  
);
```

```
ALTER TABLE student ADD CONSTRAINT student_klas FOREIGN KEY student_klas (klas_id)  
REFERENCES klas (klas_id);
```

CONSTRAINT 2: FOREIGN KEY

- Een foreign key kan specificeren wat er moet gebeuren met het child record bij een delete/update van het parent record

```
[CONSTRAINT [symbol]] FOREIGN KEY  
  [index_name] (col_name, ...)  
  REFERENCES tbl_name (col_name,...)  
  [ON DELETE reference_option]  
  [ON UPDATE reference_option]
```

reference_option:

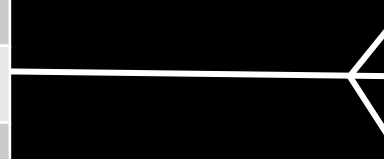
RESTRICT | **CASCADE** | **SET NULL** | **NO ACTION** | **SET DEFAULT**

klas

klas id	naam	aantal
1	Bi1a	13
2	Bi1b	13
3	Bi1c	14
4	Bi1d	12

student

stud id	Voornaam	klas id
496810	Gonny	4
513767	Henk	4
481236	Margo	1
508953	Martijn	



OPDRACHT

Wat gebeurt er bij SET NULL?



```
[CONSTRAINT [symbol]] FOREIGN KEY  
    [index_name] (col_name, ...)  
REFERENCES tbl_name (col_name, ...)  
[ON DELETE reference_option]  
[ON UPDATE reference_option]
```

reference_option:

RESTRICT | CASCADE | SET NULL | NO ACTION | SET DEFAULT

CONSTRAINT 3: UNIQUE CONSTRAINT (KEY)

- Unique constraint dwingt uniciteit af
- Kan meerdere keren in een tabel voorkomen
- Iedere waarde in een kolom is uniek
- Niet altijd ingevuld

stud_id	voornaam	tussenv	achternaam	Email
472020	Pietje		Puk	
510623	Gonny		Henkes	
496271	Rick	van	Kessel	

Uniek voor alle
studenten van de HAN

CONSTRAINT 3: UNIQUE KEY

```
create table student
(
  student_nr      int      primary key,
  voornaam        varchar(11) not null,
  tussenvoegsels  varchar(10) null,
  achternaam      varchar(20) not null,
  geb_datum       date      null,
  woonplaats      varchar(20) null,
  email           varchar(25) null,
  mobiel          varchar(15) null,
  klas_id         int       null,
  slb_id          int       null,
  constraint AK_0
    unique (email)
);
```

VRAAG:

- Wat is het verschil tussen een **primary key** constraint en een **unique** constraint?

VRAAG

- Wat is het verschil tussen een **primary key** constraint en een **unique** constraint?
 - Een primary key is not null (altijd ingevuld)
 - Er is (vaak) maar een primary key per tabel
 - Geen verwijzing naar andere tabel

CONSTRAINT 4: NOT NULL

- Dwingt af dat iets ingevuld moet zijn
- Zoals bijvoorbeeld een voornaam

CONSTRAINT 4: NOT NULL

```
create table student
(
    student_nr      int      not null
    primary key,
    voornaam        varchar(11) not null,
    tussenvoegsels  varchar(10) null,
    achternaam      varchar(20) not null,
    geb_datum       date      null,
    woonplaats      varchar(20) null,
    email           varchar(25) null,
    mobiel          varchar(15) null,
    klas_id         int      null,
    slb_id          int      null,
    constraint AK_0
    unique (email)
);
```

CONSTRAINT 5: CHECK

- Dwingt af dat er een bepaalde waarde is opgenomen

VOORBEELD VAN EEN CHECK CONSTRAINT

- Voorkom invoer niet integere gegevens

```
alter table student add constraint protect_geb_datum  
check (geb_datum>'1970-01-01' and geb_datum< '2008-01-01');
```

CONSTRAINTS INLINE BIJ CREATIE VAN TABEL

```
create table werknemer (  
  id int primary key,  
  voornaam int not null,  
  salaris decimal (10,2)  
    check (salaris>0 and salaris<10000),  
  email varchar(20) unique,  
  manager_id int,  
  constraint FK_manager foreign key (manager_id)  
    references werknemer(id)  
);
```

Primary Key

Not null

Unique

Check

Foreign Key

DE VIJF CONSTRAINTS

Naam	Afk	Betekenis
Primary Key	PK	Identificeert een rij
Foreign Key	FK	Verwijst naar een andere rij
Unique	UK	Uniek in die kolom
Not null	NN	Mag niet leeg zijn
Check	C	Controle op inhoud

SAMENVATTING

- DDL, Data Definition Language (or Statements) is de groep SQL statements waarmee je tabellen kunt aanmaken en verwijderen (CREATE, ALTER, DROP)
- Tabellen bestaan uit kolommen met verschillende datatypes (o.a. varchar, decimal en date)
- Een kolom kan een extra eigenschap hebben:
 - controle op uniciteit (unique)
 - controle op inhoud (check)

AGENDA

- Het Relationele Model
- SQL IDE's
- DDL: creatie tabellen
 - create, alter, drop
- DML: wijzigen inhoud tabellen
 - insert, delete, update
- DRL: ophalen gegevens uit tabellen
 - select

INSERT

```
insert into student values  
  (534845, 'Gonny', null, 'Henkes');
```

Of

```
insert into student  
(student_nr, voornaam, achternaam)  
values (534845, 'Gonny', 'Henkes');
```

student_nr	voornaam	tussenv	achternaam
534845	Gonny		Henkes

SYNTAX VAN HET INSERT STATEMENT

```
INSERT INTO tbl_name (column1, column2,...)  
VALUES(value1, value2,...);
```

DELETE

```
delete from student;
```

maar dit zal alle studenten verwijderen
beter is dus

```
delete from student  
where student_nr = 527727;
```

student_nr	voornaam	tussenv	achternaam

SYNTAX VAN HET DELETE STATEMENT

```
DELETE FROM  
    table_name  
[WHERE condition];
```

UPDATE

student_nr	Voornaam	tussenv	achternaam	email
514697	Thijn	van	Kempen	

```
update student
set      email = 'Thijn.vanKempen@han.nl'
where    student_nr = 514697
```

student_nr	Voornaam	tussenv	achternaam	email
514697	Thijn	van	Kempen	Thijn.vanKempen@han.nl

SYNTAX VAN HET UPDATE STATEMENT

```
UPDATE table  
SET column = expression  
  [, column =  
    expression      , ...]  
[WHERE condition];
```

VRAGEN?

AGENDA

- Het Relationele Model
- SQL IDE's
- DDL: creatie tabellen
 - create, alter, drop
- DML: wijzigen inhoud tabellen
 - insert, delete, update
- DRL: ophalen gegevens uit tabellen
 - select

SELECT

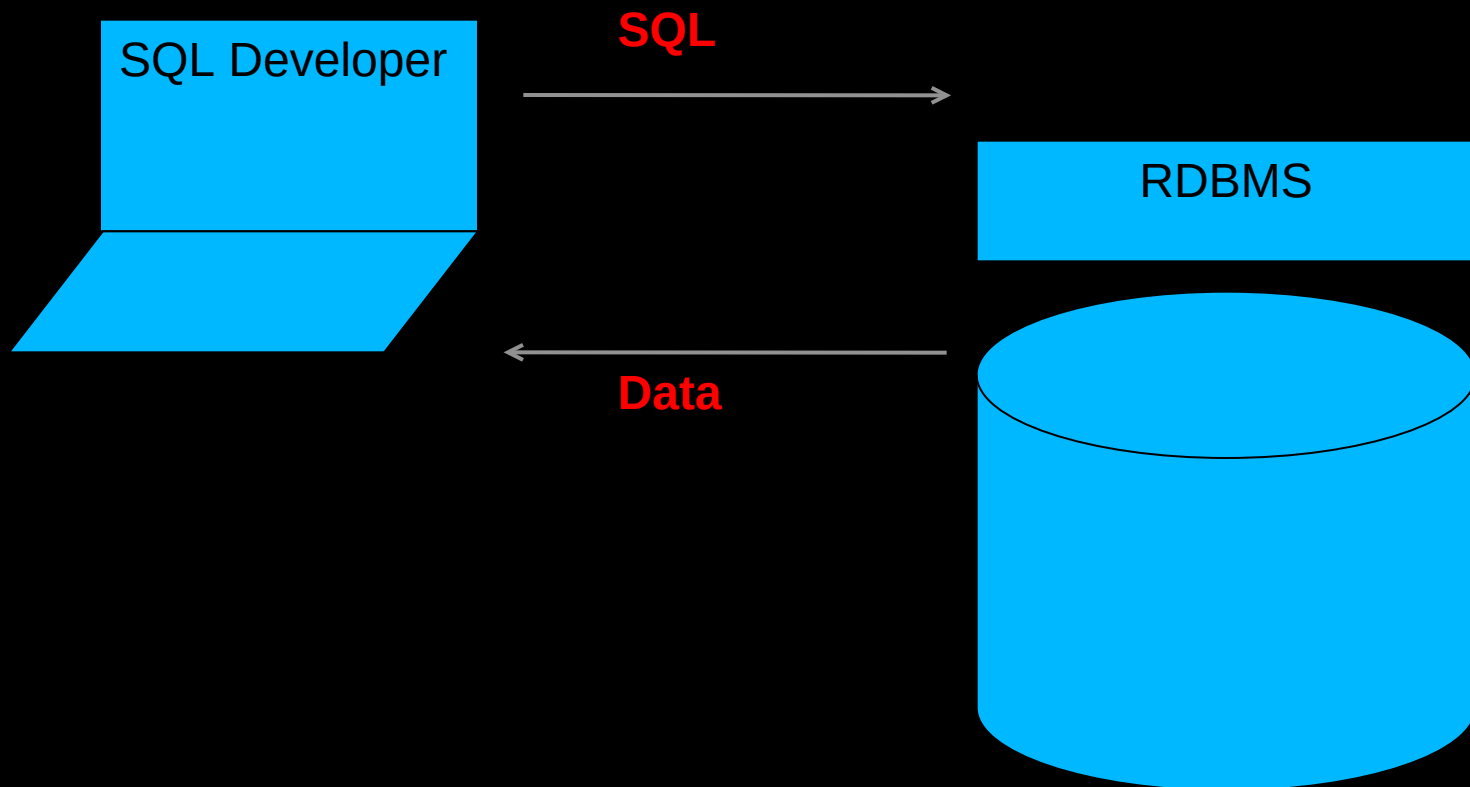
- DRL data retrieval language is het statement dat we het meest zullen gebruiken
- Het is ook het meest complexe statement
- Het begint altijd met select
- De volgende hoofdstukken zullen daar over gaan

SYNTAX VAN HET SELECT STATEMENT

```
select kolommen  
from tabel
```

Groep	Doel	Statements
DRL	Data Retrieval	select
DML	Data Manipulation	update delete insert
DDL	Data Definition	create alter drop
TCL	Transaction Control	commit rollback
ACL	Access Control	grant revoke

WERKEN MET EEN RELATIONELE DATABASE



HET SELECT STATEMENT

```
select *  
from student
```

Haal de kolommen op

uit de tabel studenten

HET GENERIEKE SELECT STATEMENT

```
select <kolomnamen | *>  
from   <tabel>
```

MEERDERE KOLOMMEN

```
select  voornaam  
        ,      achternaam  
        ,      email  
from    student
```

Inspringen is niet nodig, maar het maakt het wel beter leesbaar

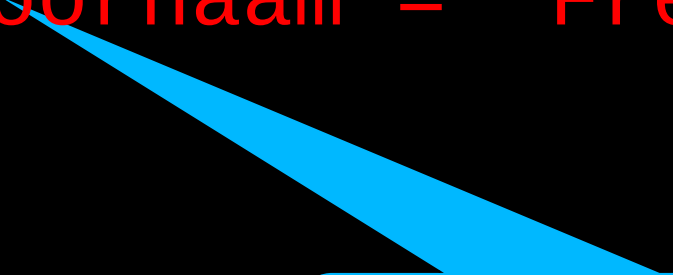
DISTINCT

```
select distinct klas_id  
from student
```

Distinct zorgt voor unieke waarden in de resultaatset

DE WHERE CLAUSULE

```
select *  
from student  
where voornaam = 'Freek'
```



De where clause legt een filter op de rijen

VERGELIJKINGS OPERATOREN

Operator	Betekenis
=	Gelijk aan
>	Groter dan
<	Kleiner dan
>=	Groter of gelijk aan
<=	Kleiner of gelijk aan
!= en <>	Niet gelijk aan

DE ORDER BY CLAUSULE

```
select *  
from student  
order by achternaam
```



De order by legt een sortering aan op de resultaatset

ORDER BY

- De volgorde is standaard ASC (default)
 - ASC - Ascending, laag naar hoog
 - DESC - Descending, hoog naar laag

SAMENVATTING SELECT STATEMENT

```
select      [distinct] <kolomnamen|*>  
from        <tabel>  
[where      <conditie>]  
[order by   <kolomnaam|getal>[ASC|DESC]]
```

VRAGEN?

SAMENVATTING

- Data Retrieval statements beginnen altijd met select
- Het select statement is het meest uitgebreide statement in SQL

BELANGRIJKE BEGRIPPEN

- Constraints
 - PK
 - FK
 - NN
 - U
 - C
- Null
- Driewaardige logica
- select
- from
- where
- order by
- update
- delete
- insert

OPDRACHT

- Maak 2 tabellen waarin klassen en studenten kunnen worden opgeslagen
- Geef iedere tabel minimaal 3 attributen
- Breng keys aan zodat parent en child records (1:n) naar elkaar verwijzen
- Vul de tabellen zodat beide tabellen een aantal records hebben
- Zorg dat studenten worden verwijderd als de klas waarin ze zitten wordt verwijderd
- Verwijder een klas. Werkt het?

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