Inhoud

[Begrippen 2](#_Toc387920247)

[DMS 3](#_Toc387920248)

[Datatypen 4](#_Toc387920249)

[Primary key: 5](#_Toc387920250)

[Foreign key: 5](#_Toc387920251)

[Naamgevingsconventies 6](#_Toc387920252)

[Aanlevering 7](#_Toc387920253)

[Data dictionary 8](#_Toc387920254)

# Begrippen

**Column/Field:**

Een column of field is een data item in de database waar alle waarden van de rows of records netjes onder elkaar staan.

**Row/Record:**

Een row of record is een data item in een database. Deze staan onder elkaar in columns of fields en meestal op volgorde van het item ID.

# DMS

**Data Management systems**

Een database management systeem (DMS) is een programma waarmee een of meer computergebruikers toegang tot de gegevens in een database kunnen creëren. De DMS beheert verzoeken van gebruikers (en verzoeken van andere programma's) zodat gebruikers en andere programma's begrijpen waar gegevens zich bevinden in de database. Visual studio gebruikt als DMS SQL, ODBC, Access en Oracle.

Hier is een lijstje van veelgebruikte DMS:

* CKAN
* Quandl
* InfoChimps
* DataMarket
* ScraperWiki
* BuzzData
* Kasabi
* Factual
* GeoIQ
* Windows Azure MarketPlace
* Avoiding Mass Extinctions Engine
* PANDA project
* Socrata

# Datatypen

Boolean: wordt gebruikt om bij he houden of dat iets waar of niet waar is.

Byte: 1 byte in het geheugen die gebruikt wordt om waardes tussen 0 en 255 op te slaan.

Integer: wordt gebruikt om gehele getallen op te slaan.

Short int: wordt gebruikt om gehele getallen op te slaan, maar is kleiner dan een int.

Long: wordt gebruikt om gehele getallen op te slaan, maar is groter dan een int.

Float: wordt gebruikt om kommagetallen op te slaan.

Double: wordt gebruikt om kommagetallen op te slaan, is groter dan een float.

Decimal: wordt gebruikt om kommagetallen op te slaan. Decimals zijn nauwkeuriger dan floats

Char: wordt gebruikt om een karakter op te slaan.

String: een array van Chars.

Nchar: Een char array van n aantal chars.

Nvarchar: Een char array van n aantal chars met een max waarde voor de hoeveelheid gebruikte opslagruimte.

Primary key: Wordt gebruikt om het ID van een item op te slaan. Elke primary key is uniek.

Foreign key: De foreign key zorgt ervoor dat als gegevens in een record verwijzen naar gegevens in een ander record, dat er alleen verwezen kan worden naar gegevens die ook echt bestaan.

Time: Word een tijd in opgeslagen.

Date: Word een datum in opgeslagen.

# Primary key:

De primary key of unique key wordt gebruikt in relationele databases. Een van de belangrijkste regels van het relationeel model is dat alle rijen in een tabel te identificeren zijn door middel van de primary key. De primary key is dus altijd uniek.

Een primary key werkt als een soort hub die leidt naar meer informatie. Hij wordt gebruikt om rijen in tabellen te identificeren en om gegevens van verschillende tabellen aan elkaar te koppelen.

# Foreign key:

De foreign key is een van de meest gebruikte beperkingen in databases. De foreign key zorgt ervoor dat als gegevens in een record verwijzen naar gegevens in een ander record, dat er alleen verwezen kan worden naar gegevens die ook echt bestaan.

# Naamgevingsconventies

**Naamgeving**

* Tabellen krijgen voor de naam tbl met een underscore en de meervoud van wat deze tabel heeft. Op deze manier is het duidelijk dat het om een tabel gaat en wat er in staat.  
  Voorbeeld: tbl\_teams, tbl\_spelers
* Elke kolom heeft duidelijke benamingen voor de variabelen en de ID als eerste staan. De namen hiervan staan in meervoud omdat er meerdere variabelen in staan.
* Elke rij krijgt een duidelijke naam in enkelvoud omdat het over variabelen gaat.
* Als 2 tabellen met elkaar verbonden worden gebruiken we de ID.  
  Voorbeeld: Als tbl\_spelers word verbonden met tbl\_teams dan is de verbonden variabel teams\_ID in tbl\_spelers.

# Aanlevering

We kunnen de tabellen op meerdere manieren binnen krijgen zoals: XML, MySQL, Microsoft SQL server, JSON of AJAX.

We zijn uitgegaan vanuit MySQL en hebben van daaruit vergeleken met een aantal andere aanlever methoden waaronder JSON. JSON is een taal die met javascript werkt en kan dienen als een goede vervanger van XML of AJAX. Het nadeel van JSON zijn de beveiligingsissues, want doordat het met javascript werkt is het relatief gemakkelijk om code te importeren.

Een van de andere manieren van aanleveren is met Excel, maar dit was al snel verworpen omdat Excel niet gemakkelijk meerdere gebruikers ondersteunt en vastloop problemen vertoond wanneer je dit wel voorelkaar hebt gekregen.

XML is een van de methoden waarvan al enige ervaring is binnen de projectgroep. XML is goed leesbaar en erg licht om te draaien. In XML is het alleen niet makkelijk om tabellen te maken en wordt het snel trager naarmate de hoeveelheid van doorgestuurde data toeneemt.

Het laatst hebben we nog gekeken naar AJAX, maar dit heeft hetzelfde beveiligingsprobleem als JSON, omdat het ook gebruik maakt van javascript. Bovendien heb je een externe database nodig waar AJAX zijn data uithaalt, waar MySQL zijn eigen database heeft.

Gekeken naar het onderzoek dat we hebben gedaan is de beslissing genomen om MySQL te gebruiken. Het is een lichte snelle manier van data leveren zonder beveiligingsproblemen. Bovendien heeft MySQL zich op professioneel gebied al eerder bewezen bij meerdere financiële en regeringsinstanties zoals de ‘Bank of Finland’ waar het gebruikt wordt om simulaties te draaien en NASA gebruikt het in hun ‘NASA Jet Propulsion Lab’.

# Data dictionary

Een data dictionary kan worden gezien als een geautomatiseerd documentatiesysteem of als een woordenboek betreffende de definitie van alle entiteittypen, attributen en relaties. Afhankelijk van het specifieke product kunnen tevens de views of subschema's, de fysieke database kenmerken, de gebruikers, hun privileges en de programma's worden gedefinieerd. De data dictionary is zelf ook een database.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tabel: tbl\_teams** | Type | Auto inc | Null | Uitleg |
| ID | Primary key | Auto Inc. | NOT NULL | Primary ID met sleutel |
| Teams | VarChar 25 |  | NOT NULL | Team naam |
| Doelpunten | Int 2 |  |  | Aantal doelpunten per team |
| Gewonnen | Int 2 |  |  | Aantal keer gewonnen |
| Verloren | Int 2 |  |  | Aantal keer verloren |
|  |  |  |  |  |
| **Tabel: tbl\_wedstrijden** | Type | Auto inc | Null | Uitleg |
| ID | Primary Key | Auto Inc. | NOT NULL | Wedstrijd ID |
| Teams\_ID | Int 2 |  | NOT NULL | Team ID, van tbl\_teams |
| Tijden | Time |  | NOT NULL | Wedstrijd tijd |
| Zaal | VarChar 25 |  | NOT NULL | De zaal van de wedstrijd |
| Doelpunten | Int 2 |  |  | Aantal doelpunten per wedstrijd |
|  |  |  |  |  |
| **Tabel: tbl\_spelers** | Type | Auto inc | Null | Uitleg |
| ID | Primary Key | Auto Inc. | NOT NULL | Speler ID's |
| Naam | VarChar 25 |  |  | Speler namen |
| Leeftijd | Int 2 |  |  | Speler leeftijden |
| Geslacht | Default = Man | VarChar 5 |  | Speler geslacht |
| Teams\_ID | Int 2 |  |  | Teams ID van tbl\_teams |
| Punten | Int 2 |  |  | Aantal doelpunten per speler |