|  |
| --- |
| BAZEN BV. |
| Zaalvoetbal Toernooi |
| Verslag Database |
|  |
| **Mathijs Arts, Geert Cocu, Boaz Frey, Tommy de Hoon** |
| **22-4-2014** |

|  |
| --- |
|  |



Inhoud

[Inleiding 2](#_Toc388279391)

[Waarom een database? 2](#_Toc388279392)

[Wat moet er in onze database komen? 2](#_Toc388279393)

[Wat voor database? 3](#_Toc388279394)

[Begrippen 3](#_Toc388279395)

[Column/Field 3](#_Toc388279396)

[Row/Record 4](#_Toc388279397)

[Database Management System (DBMS) 4](#_Toc388279398)

[Datatypes 4](#_Toc388279399)

[Primary Key 4](#_Toc388279400)

[Foreign Key 4](#_Toc388279401)

# Inleiding

Dit verslag is geschreven door Database Manager Geert Cocu.

Met dit verslag wordt duidelijk gemaakt waarom wij een database nodig hebben, en hoe wij deze gebruiken. Ook wordt er in omschreven hoe wij deze database koppelen aan onze applicatie.

# Waarom een database?

Een database is handig als je veel gegevens wilt opslaan waar je vaak bij zou moeten komen.

Zoals bij dit project het geval is, moeten wij een database maken met daarin de teams die in de wedstrijden spelen, met welke spelers daarin spelen. Ook moeten we de eindscores van afgelopen wedstrijden.

Wat ook handig is aan een database, je kan deze online hosten, zodat als mensen het programma gebruiken met een database, er altijd bij kunnen. En mocht er dan een update in de database vinden, krijgt iedere gebruiker de update meteen door. En hoeft niet iedere applicatie geüpdate worden met de vernieuwde database.

# Wat moet er in onze database komen?

In deze database moeten de volgende dingen in komen:

**De wedstrijden**

* Wedstrijd schema
* De teams
* De spelers
* De eindscore van iedere wedstrijd
* Hoeveel punten ieder team heeft

**De voorspellers**

* Voorspelling welke team wint
* Gewonnen ja/nee, en krijgen een punt als ze winnen
* Wedstrijd geschiedenis
* Hoeveel punten ze hebben
* Gebruikersnaam en wachtwoord

# Wat voor database?

Je hebt veel verschillende talen waarin databases worden geschreven, zoals: SQL, XML en Json.

## SQL

SQL staat voor Structured Query Language. Het staat voor een structurele taal die voornamelijk gebruikt wordt in databases. SQL communiceert met een database doormiddel van een “zoek query”. Met deze query wordt er gezocht in de database en stuurt het gegevens terug naar de client. De voordelen van SQL zijn: het is redelijk snel, en krijgt resultaten snel terug, en je hoeft niet veel te programmeren om een SQL database te laten werken in een programma. Maar de nadelen zijn: Dat het opzoeken vrij lastig is met SELECT en FROM enzovoorts.

## XML

XML staat voor Extensible Markup Language. Het is een structurele taal die wordt gebruikt in een XML-bestand. De opbouw van XML is te vergelijken met HTML. Het maakt gebruik van tags. Deze tags kan je een naam geven waardoor je een element maakt. Deze elementen of nodes kunnen tekst bevatten of andere elementen of nodes. Dit heten de childnodes. De nadelen zijn dat een node altijd een string terug geeft. Bij een int zou je eerst je variabelen moeten parsen voordat je het wilt gebruiken.

## JSON

JSON staat voor JavaScript Objec Notation. JSON wordt veel gebruikt om data tussen een server en een applicatie uit te wisselen, als een alternatief voor XML. JSON wordt dus veel gebruikt in databases. JSON is kleiner en sneller in vergelijking met XML. Maar JSON heeft geen schema ondersteuning, wat XML weer wel heeft.

## Onze keuze

We gebruiken een SDF Database omdat deze gebruikt wordt in Visual Studio, we hier voornamelijk les over krijgen, en omdat SDF vrij makkelijk te gebruiken is in C#. En omdat we er al eerder mee hebben gewerkt.

# Begrippen

Hier worden er een paar begrippen uitgelegd die veel gebruikt gaan worden in databases, en dus ook in ons eigen database.

## Column/Field

Een column, of field, is een kolom met data *onder* () elkaar. Voorbeeld: Een kolom wordt gebruikt om bij iedereen aan te geven wat de voor- en achternaam is.

|  |  |  |
| --- | --- | --- |
| **ID** | **Voornaam** | **Achternaam** |
| 1 | Henk | Stoepert |
| 2 | Jan | v. Enkel |

## Row/Record

Een Row, of een record, is een rij met data *naast (*) elkaar. Voorbeeld: Een rij wordt gebruikt om het ID van iedere persoon aan te geven.

|  |  |  |
| --- | --- | --- |
| **ID** | **Voornaam** | **Achternaam** |
| 1 | Henk | Stoepert |
| 2 | Jan | v. Enkel |

## Database Management System (DBMS)

Een DBMS wordt gebruikt om een database met zijn gegevens te bewaken en te beheren. Een database bestaat uit 3 onderdelen: De gegevens die er in zijn opgeslagen, een programma waarin je de gegevens kunt onderhouden (DBMS), en een programma met een UI (cliënt) om de gegevens te gebruiken.

## Datatypes

Iedere column in een database heeft een naam en een datatype. Met een datatype wordt bedoelt wat voor data je in die column zet. Bijvoorbeeld, als je als datatype *boolean* zet, kan je in die column alleen “ja” of “nee” invullen. Dit voorkomt dat je verschillende soorten gegevens in een column invult.

## Primary Key

De primary key, is de sleutel die je aan de hoofdtabel in je database hangt. De primary key moet:

* Uniek zijn. De waarde van de tabel is uniek, en kan geen 2e gelijke tabel zijn.
* Niet leeg zijn. De tabel kan niet leeg zijn, en moet gegevens bevatten
* Relaties tussen tabellen leggen. Zodat tabellen niet een omweg hoeven te maken langs andere tabellen.

## Foreign Key

Een foreign key, of een verwijzende sleutel, is een sleutel die vanuit een table naar een andere tabel kan verwijzen. Zo kan je bijvoorbeeld in een table van teams verwijzen naar een table met spelers.

# Naamgevings conventie

|  |  |
| --- | --- |
| Conventie naam | Uitleg |
| Taal | Alle tabellen in het Engels |
| Hoofdletter | Iedere inhoud van een hokje begint met een hoofdletter |
| Lengte inhoud | De inhoud van iedere hokje mag niet langer zijn als 15 karakters |
| Primary key | Gebruiken voor een unieke waarde, zoals het ID van de bieders |
| Foreign key | Gebruiken als je gegevens met elkaar moet koppelen, zoals de spelers en hun team. |