



FACULTEIT WETENSCHAPPEN
Vakgroep Zuivere Wiskunde en Computeralgebra

Academiejaar 2005–2006

Centraal evaluatie- en rapporterings- systeem voor een middelgrote school

Ruben CAPIAU
Hendrik VAN DE MOORTELE

Promotor:
dr. J. DE BEULE

Co-promotor:
prof. dr. L. STORME

Scriptie voorgelegd aan de Faculteit Ingenieurswetenschappen,
voor het behalen van de academische graad van
MASTER IN DE TOEGEPASTE INFORMATICA

Dankwoord

Graag zouden we onze promotor, Jan De Beule, willen danken voor de goede begeleiding van deze scriptie. Bedankt voor de documentatie, de raadgevingen en tips en voor het opvolgen van onze vorderingen.

Ook willen we de mensen van de academie in Deinze bedanken voor de hartelijke ontvangst tijdens ons eerste kennismakende gesprek. Dank aan directeur Geert Dhondt om deze opdracht aan ons toe te vertrouwen. En *last but not least*, dank aan Stefaan Cornelus om steeds onze mails met vragen te beantwoorden en voor het toeleveren van de nodige data.

De auteurs geven de toelating deze scriptie voor consultatie beschikbaar te stellen en delen van de scriptie te kopiëren voor persoonlijk gebruik. Elk ander gebruik valt onder de beperkingen van het auteursrecht, in het bijzonder met betrekking tot de verplichting de bron uitdrukkelijk te vermelden bij het aanhalen van resultaten uit deze scriptie.

6 juni 2006,

Ruben Capiou

Hendrik Van de Moortele

Overzicht

TITEL:

Centraal evaluatie- en rapporteringssysteem voor een middelgrote school

AUTEURS:

Ruben Capiou

Hendrik Van de Moortele

PROMOTOR:

dr. J. De Beule

CO-PROMOTOR:

prof. dr. L. Storme

Scriptie afgewerkt aan de Faculteit Wetenschappen, Vakgroep Zuivere Wiskunde en Computeralgebra, Universiteit Gent

Scriptie voorgelegd aan de Faculteit Ingenieurswetenschappen (Universiteit Gent), voor het behalen van de academische graad van Master in de Toegepaste Informatica

Samenvatting

In het kader van deze scriptie werd de databank van de Stedelijke Academie voor Muziek, Woord en Dans Deinze — die ondersteuning moet bieden aan een centraal evaluatie- en rapporteringssysteem — volledig opnieuw ontworpen. De databank werd compacter gemaakt en aan de hand van een Entity/Relationship-model zodanig geconcipieerd dat het systeem flexibeler werd en meer mogelijkheden kreeg. De bijhorende website, die leerkrachten moet toelaten om online studenten te evalueren en de rapporten af te drukken, werd eveneens volledig herwerkt.

Trefwoorden: evaluatie- en rapporteringssysteem, MySQL, PHP, LaTeX, relationele databank

Inhoudsopgave

1	Inleiding	1
2	Opdrachtafbakening en probleemdefinitie	3
2.1	Analyse van het oorspronkelijke systeem	3
2.2	Vereisten gesteld aan het nieuwe systeem	5
2.3	Chronologie van uitvoering en systeemstructuur	6
3	Theoretische achtergrond	8
3.1	De scripttaal PHP	8
3.1.1	Historiek en eigenschappen	8
3.1.2	Syntax en gebruik	9
3.1.3	Vaak gebruikte functies	9
3.2	De database MySQL	16
3.2.1	Theoretische achtergrond	17
3.2.2	De verschillende MySQL uitdrukkingen	17
3.2.3	Gevaarlijk probleem bij gebruik PHP en MySQL	20
3.3	De LaTeX markuptaal	21
3.3.1	Opbouw en tags	21
4	Bespreking van de oplossing op technisch niveau	24
4.1	Ontwerp van de databank	24
4.1.1	Toelichting bij de ontwerpmethodologie	24
4.1.2	Opstellen van het relationeel databankmodel	26
4.2	De databankbeheerpagina's	43
4.2.1	Overzicht van de beheerpagina's	43
4.2.2	De centrale beheerpagina	45
4.2.3	Wachtwoord	46
4.2.4	Beheer_ cursist	47
4.2.5	Beheer_ leerkracht	47
4.2.6	Beheer_ evaluatie	47
4.2.7	Rapport	47
4.2.8	Het uploaden van de gegevens	48
4.2.9	Het aanmaken van de tabellen	49
4.3	De nieuwe gebruikerswebsite	52

4.3.1	Structuur van en navigatie binnen de website	52
4.3.2	Authenticatie en andere beveiligingsaspecten	54
4.3.3	Welkompagina en generieke look van de website	56
4.3.4	Weergave van de klasgroepen en van de studenten	59
4.3.5	Invoerrechten en -pagina	63
4.3.6	Controle van de input en invoer in de databank	65
4.3.7	Wijzigen of bekijken van de gegevens per student	67
5	Besluit	71
6	Bibliografie	72

Hoofdstuk 1

Inleiding

De Stedelijke Academie voor Muziek, Woord en Dans Deinze mag zich met zijn ca. 1800 leerlingen tot de grotere academies van Oost-Vlaanderen rekenen. Om dit cijfer in perspectief te plaatsen: het gemeentelijk deeltijds kunstonderwijs in onze provincie telde vorig jaar 20 academies die deze drie studierichtingen aanbieden, goed voor 20987 leerlingen in het totaal. De Deinse academie kende vanaf het midden van de jaren '60 een uitbreiding onder de vorm van filialen in Zulte en Kruishoutem. Dankzij de inspanningen van Geert Dhondt, die sinds 1994 het directeurschap waarneemt, konden in 2000 en 2002 ook onderafdelingen in St.-Martens-Latem en Nazareth opgericht worden. Vandaag telt de academie meer dan 65 leerkrachten en personeelsleden, dankzij wiens inzet een waaier van 48 opleidingen kan aangeboden worden.

Tijdens de voorbije jaren werd door Stefaan Cornelus, leerkracht klarinet en saxofoon aan de academie, een publieke website uitgebouwd. Daarnaast werd door hem ook een centraal evaluatiesysteem op poten gezet, waarop leerkrachten kunnen inloggen om hun leerlingen te beoordelen. Hierna kan een rapporteringsfase gestart worden, zodanig dat op elk evaluatiemoment (Kerst, Pasen en het einde van het jaar) een automatisch gegenereerd rapport aan de leerlingen kan worden meegegeven. Deze aparte website maakt gebruik van de klassieke ingrediënten: HTML gecombineerd met PHP-scripts die een MySQL-server aanspreken en Cascading Style Sheets (CSS) voor de algemene opmaak.

De talrijke mogelijkheden bij inschrijving (verschillende graden, opties en vakken), de unieke koppeling tussen leerkrachten en de individuele klasgroepen op de vijf afdelingen en het groot aantal variabelen in het systeem maken het ontwerp van een ondersteunende databank tot een serieuze denkoefening. Hoewel het oorspronkelijke systeem zich goed van zijn taak kweet, bleken er toch beperkingen verbonden te zijn aan de gebruikte manier van werken. Vooral de complexiteit van de databank stond niet in verhouding tot de vereisten die eraan gesteld werden. Ook was er vanuit de academie de wens om meer flexibiliteit in het systeem in te bouwen (hierover meer in hoofdstuk 2). Het doel van deze scriptie was dan ook om deze beperkingen te elimineren en om van de gelegenheid gebruik te maken om de databankstruc-

tuur te vereenvoudigen. Andere belangrijke zaken, zoals de vormgeving van de website, de mogelijkheid om gegevens makkelijk in te voeren en te back-uppen en beveiligingsaspecten werden daarbij niet uit het oog verloren. Een grondige herstructurering van de databank impliceerde wel dat de oorspronkelijke PHP-scripts niet meer gebruikt konden worden. Er werd dus van een schone lei gestart.

Wat kunt u in deze scriptie verwachten? In *hoofdstuk 2* wordt er dieper ingegaan op de beperkingen van het systeem die verholpen dienden te worden. Er wordt m.a.w. een soort ‘requirement analysis’ uitgevoerd, om het in het informaticajargon uit te drukken. In *hoofdstuk 3* lichten we de gebruikte talen en systemen toe, zonder echt in detail te treden. Met dit hoofdstuk willen we de lezers die niet vertrouwd zouden zijn met deze talen de nodige achtergrond bieden. *Hoofdstuk 4* vormt het belangrijkste deel van deze scriptie. Hier zetten we de ideeën die we uitgewerkt hebben op papier. Meer bepaald komt het ontwerp van de nieuwe databank, de structuur en functionaliteit van de website en de verwerking van de rapporten aan bod. Tot slot volgt ons besluit.

Hoofdstuk 2

Opdrachtafbakening en probleemdefinitie

2.1 Analyse van het oorspronkelijke systeem

Men kan het oorspronkelijke systeem (evenals het nieuwe systeem) opdelen in een viertal eenheden, die apart van elkaar dienen behandeld te worden bij het ontwerp van het systeem maar die elk een cruciale rol innemen binnen het geheel van het evaluatiesysteem. Dit zijn met name: de databank, de scripts voor het beheer en het manipuleren van de databank, de scripts verantwoordelijk voor de functionaliteit van de gebruikerswebsite en ten slotte de ‘stylesheet’, een CSS-bestand dat instaat voor de lay-out en de overkoepelende vormgeving van de volledige website. Het is de mate waarin deze vier subsystemen op elkaar aansluiten en van elkaars mogelijkheden gebruik maken die bepaalt hoe goed het uiteindelijke systeem beantwoordt aan de verwachtingen van de gebruikers.

In het systeem zoals het tot nu toe gebruikt werd, schulde het probleem hoofdzakelijk in de *databank*. Ten gevolge van de verschillende vereisten waaraan de databank moest voldoen (met het oog op de bevraging door de scripts), werd deze tijdens de ontwikkeling van het systeem vermoedelijk steeds verder aangevuld met nieuwe tabellen, onder andere om de relatie tussen de diverse tabellen vast te leggen. Er werd dus van bij het begin gewerkt volgens een soort relationeel databankmodel. Doordat de structuur van dit model echter niet volledig vast lag op voorhand (wat begrijpelijk is, aangezien men niet altijd kan voorzien waar men met het systeem naar toe wil), diende de structuur en de functionaliteit mettertijd in de databank aangebracht te worden, wat moeilijk of op den duur praktisch onhaalbaar was.

Twee voorbeelden illustreren dat de databank conceptueel niet zo goed in elkaar zat:

- in de tabel met de administratieve gegevens van de studenten waren per student een zestal ‘sleutels’ aanwezig die het verband moesten leggen met andere tabellen of die

bevragingen vanuit de scripts mogelijk moesten maken. Deze sleutels bestonden niet uit getallen (zoals de gebruikelijke ‘identifiers’) maar uit vrij ingewikkelde letter- en cijfercombinaties (bijvoorbeeld bestaande uit een concatenatie van de afkorting van het vak, de initialen van de leerkracht, het leerjaar en de klasgroep en één of meerdere letters die de afdeling aangaven). Deze zes codes (in concreto: de attributen ‘optiekort’, ‘klasgroep’, ‘sectie’, ‘graad’, ‘code’ en ‘leerjaar’) bevatten vrij veel overlap qua informatie en waren bovendien niet echt consequent opgebouwd (al naargelang werd ‘Deinze’ in het attribuut ‘code’ bijvoorbeeld afgekort tot D, DE, DEI of DEIN).

- zeer opvallend bij het bekijken van de oude databank: voor elke bestaande combinatie van een vak, leerkracht, evaluatiemoment, graad en sectie (muziek, woord of dans) werd een aparte tabel in de databank aangemaakt. Dit resulteerde bijgevolg in enkele honderden tabellen voor een enkel schooljaar.

Voor de *scripts* had deze databankstructuur het positieve gevolg dat de ‘queries’ (bevragingen van de databank a.d.h.v. de Structured Query Language SQL) vrij eenvoudig konden gehouden worden. Doordat de databank echter bepaalde noodzakelijke informatie niet bevatte (zoals welke criteria er voor de verschillende vakken dienen gehanteerd te worden bij het evalueren van de leerlingen), zag men zich vaak genoodzaakt om situatieafhankelijke keuzemogelijkheden hard te coderen in de scripts, wat een verlies betekende aan flexibiliteit. De lettercodes waarvan eerder sprake was werden ook aangewend voor de communicatie naar de gebruiker toe (weergave van de klasgroepen). In de meeste gevallen beschikte de gebruiker daarmee over voldoende informatie om zijn/haar weg te vinden in de eigen klasgroepen, maar van een echt transparante weergave van de klasgroepen was er geen sprake. Verder merkten we bij het bestuderen van de databank nog een kleine beveiligingsfout op. In het oorspronkelijke systeem werd gewerkt met de HTTP-authenticatiefunctie van PHP om gebruikers te laten inloggen met hun gebruikersnaam en wachtwoord. Van het ingegeven wachtwoord werd met MD5 (‘Message-Digest algorithm 5’) de unieke hashwaarde berekend, waarna deze waarde vergeleken werd met de MD5-‘versleutelde’ wachtwoorden in de databank om te verifiëren of het een gelegitimeerde gebruiker betrof. In diezelfde databanktabel werden echter ook de niet-versleutelde wachtwoorden opgeslagen, en ook in één van de PHP-scripts kon men de wachtwoorden makkelijk terugvinden. Daar de scripts en de databank in principe enkel voor de beheerder toegankelijk zijn, vormde dit geen al te groot risico. De niet-versleutelde wachtwoorden leerden ons dat slechts 6 van de 64 geregistreerde gebruikers zich de moeite getroost hadden om het hen toegewezen wachtwoord (van de vorm 0x0y00z voor gebruiker z in het schooljaar 200x-200y) te wijzigen in een random letter- en/of cijfercombinatie, hoewel men deze wijziging zelf via de website kon doorvoeren. Een externe persoon met slechte bedoelingen die de redenering achter deze standaardwachtwoorden zou te weten komen zou m.a.w. vrije toegang krijgen tot het volledige evaluatiesysteem! De leerkrachten worden dus

best nog eens gewezen op het belang van een degelijk wachtwoord. Over de *opmaak* van de ‘oude website’ kunnen we kort zijn: degelijk en mooi.

2.2 Vereisten gesteld aan het nieuwe systeem

Tijdens een gesprek met de directeur, de systeembeheerder en de verantwoordelijke van het secretariaat maakten we kennis met het huidige systeem en leerden we ook de verwachtingen kennen betreffende het nieuwe systeem. Een eerste belangrijke voorwaarde waaraan het nieuwe systeem absoluut moest voldoen is compatibel te zijn met de huidige manier van werken op het secretariaat van de academie. Daar worden na de jaarlijkse inschrijvingen namelijk de invoerbestanden voor het webgebaseerde evaluatie- en rapporteringssysteem gecreëerd. Via een speciaal daarvoor ontworpen computerprogramma worden de studentengegevens (coördinaten, gekozen opties en vakken, ...) door de mensen van het secretariaat ingevoerd, met een groot Excel-bestand als resultaat. Hieruit worden de nodige kolommen (velden) geselecteerd door de beheerder. Er wordt vervolgens gebruik gemaakt van het ‘Comma-Separated Values’ (CSV)-bestandsformaat om deze geselecteerde Excel-gegevens in de MySQL-databank te introduceren. Aangezien dit de vertrouwde gang van zaken was gedurende de voorbije jaren, leek het logisch om deze *invoermethode te behouden* in het nieuwe systeem. Om dezelfde reden verkoos men om de leerlingen manueel een jaar hoger te steken bij het begin van een nieuw schooljaar, en dit niet door het systeem te laten doen.

Uiteraard staat het ontwerp van de nieuwe databank en de bijhorende website centraal in deze opdracht. Er was vanuit de academie de vraag om *meer flexibiliteit* in het systeem in te bouwen. Men zou namelijk de beoordelingscriteria die gehanteerd worden bij het evalueren van de leerlingen elk jaar willen kunnen aanpassen. Deze criteria zijn variabel in aantal en afhankelijk van het vak en van de graad waarin dat vak gedoceerd wordt. Ook diende er rekening gehouden te worden met het feit dat sommige vakken geen partiële examens hebben rond Pasen, andere dan weer wel en dat hierbij op verschillende manieren gequoteerd kan worden (algemeen totaal of opsplitsing van het resultaat in ‘rubrieken’). Verder worden de beoordelingscriteria bij enkele vakken (vnl. van de secties woord en dans) opgedeeld in categorieën. Deze onderverdeling moet dus ook zichtbaar zijn op de evaluatiepagina van de website en op het rapport van de leerlingen. Ook andere mogelijkheden, zoals het feit dat leerlingen vrijgesteld kunnen worden, dat leerlingen hetzelfde vak bij verschillende leerkrachten kunnen volgen tijdens hetzelfde jaar en dat leerlingen meerdere opties in verschillende graden kunnen volgen, dienen op voorhand ingecalculeerd te zijn. Wat de op de nieuwe databank gebaseerde website betreft, diende *op zijn minst dezelfde functionaliteit* als voorheen aanwezig te zijn. Waar mogelijk mocht de gebruikersinterface altijd verbeterd worden.

Ooit ging de volledige databank met alle gegevens van een volledig schooljaar verloren. De webhost die de databank op z’n servers staan had was toen niet in staat om een *reservekopie*

van de verloren gegevens ter beschikking te stellen. Ook met het herinstalleren van de eigen ‘dumps’ waren er toen problemen. Daarom moet in het nieuwe systeem de mogelijkheid voorzien worden om reservekopieën te maken van de databank, dumps die het systeem terug in z’n oorspronkelijke staat kunnen herstellen.

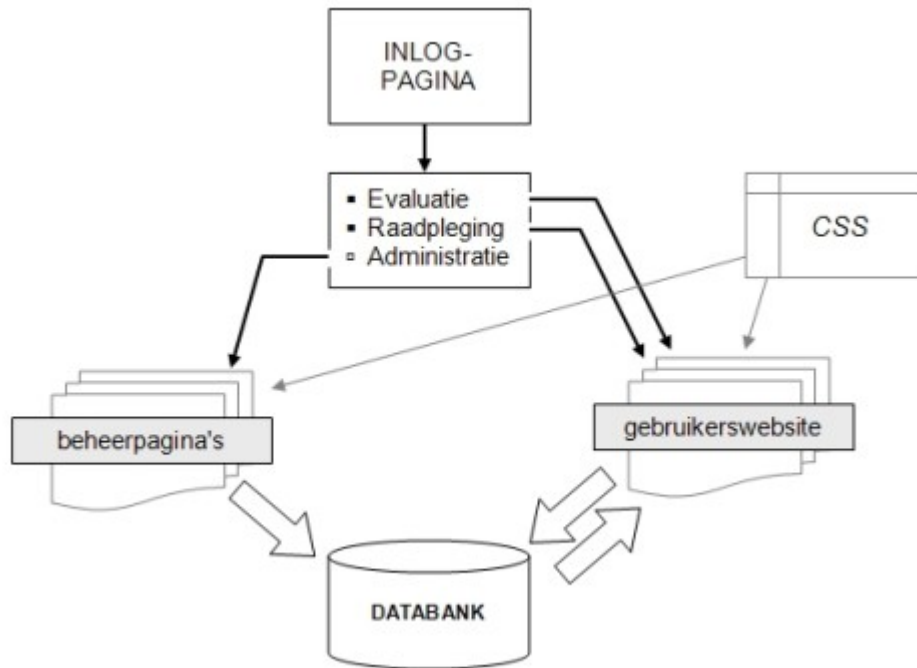
Het nogal omslachtige procédé voor het afdrukken van de rapporten ziet men ook liever verdwijnen. Het nieuwe systeem moet het mogelijk maken om de rapporten van alle leerlingen vlot af te drukken in één keer. Ook de *lay-out van de rapporten* is voor verbetering vatbaar. In het verleden had men nl. te kampen met zgn. ‘widows’ (de laatste regels tekst van een rapport, die nog net op de volgende bladzijde terechtkomen, waardoor het rapport twee pagina’s i.p.v. één pagina beslaat) en ‘orphans’ (de titel van een rubriek op het rapport staat als laatste lijn op een blad, de bijhorende tekst of cijfers staan op het volgende blad).

2.3 Chronologie van uitvoering en systeemstructuur

In een eerste fase werd de databank grondig onder handen genomen. Omdat een volledig nieuw ontwerp zich opdrong, werd gestart met het opstellen van een beschrijvende tekst die als basis moest dienen voor het latere ‘Entity/Relationship’-diagram. Uitgaande van dit diagram, dat in totaal een vijftal keer werd aangepast vooraleer te komen tot het definitieve model, werd vervolgens een relationeel schema opgesteld. Dit schema leerde ons welke tabellen (‘relaties’) de nieuwe databank zou moeten bevatten. Deze nieuwe structuur werd geïmplementeerd in MySQL: er werd een zogenaamde ‘dummy-databank’ aangemaakt, die manueel opgevuld werd met testdata. Deze testdata werden gehaald uit de input-Excelfile van het huidige schooljaar, en hierbij werd getracht om van alle mogelijke gevallen een voorbeeld in de testdatabank aanwezig te hebben.

Nu de databankstructuur vastlag, kon gestart worden met het ontwerp van de website. Er werd eerst een gedetailleerde ‘flow chart’ (grafische voorstelling van de navigatiestructuur) opgesteld van de oorspronkelijke website. Met deze chart als basis en met de nieuwe databankstructuur in het achterhoofd, werd vervolgens een schets gemaakt van hoe de nieuwe website er diende uit te zien qua structuur. In deze structuur kan men twee delen onderscheiden: enerzijds die pagina’s die gebruikt zullen worden door de beheerder bij het opvullen van de databank met nieuwe gegevens (of voor het aanpassen van bestaande data) en anderzijds de pagina’s die het evalueren van leerlingen en het raadplegen van databankgegevens door de gebruikers moeten toelaten. Deze twee delen (zie Figuur 2.1) gaven dus aanleiding tot twee aparte sets van PHP-files (deze zullen apart besproken worden). Wanneer men de URL intikt van de nieuwe website, komt men terecht op een inlogpagina. Eenmaal ingelogd, krijgt men op een centrale pagina de keuze uit twee of drie mogelijkheden (naargelang de bevoegdheden die men heeft): evaluatie, raadpleging of administratie. De eerste twee opties leiden de gebruiker naar de gebruikerswebsite, de derde optie (in principe voorbehouden voor

de beheerder) biedt een link naar de beheerpagina's. Ten slotte werd ook de opmaak van de website verzorgd. Deze werd volgens de regels van de kunst gescheiden gehouden van de HTML- en PHP-content. De website werd bedacht met een generieke 'look' om de onderlinge samenhang van de webpagina's in de verf te zetten. Hiervoor werd — zoals eerder vermeld — dankbaar gebruik gemaakt van Cascading Style Sheets.



Figuur 2.1: Globale structuur van het systeem (vereenvoudigde voorstelling).

Hoofdstuk 3

Theoretische achtergrond

De verschillende talen waarvan gebruik wordt gemaakt, worden kort toegelicht en de meest gebruikte functies besproken.

3.1 De scripttaal PHP

3.1.1 Historiek en eigenschappen

Een scripttaal is een programmeertaal waarin een script geschreven is. Dit is een klein programma om vaak voorkomende taken te automatiseren. Oorspronkelijk waren dit een reeks commando's, later werden deze uitgebreid met lussen (for, while,..), controlestructuren (if-then-else) en variabelen. Zo werden ze kleine programmaatjes. Een greep uit de verschillende scripttalen: Perl, Python, ASP, PHP,... De laatste twee zijn specifiek voor het web ontwikkeld. We gebruiken in deze thesis PHP. In 1994 ontwierp Rasmus Lerdorf deze taal. Oorspronkelijk stond PHP voor Personal Home Page, tegenwoordig staat het voor PHP: Hypertext Preprocessor.

Het is een server-side scripttaal. Dit wil zeggen dat de verwerking gebeurt op de server. Dit in tegenstelling tot client-side talen zoals javascript, welke door de browser worden verwerkt. PHP kan zowel op zich gebruikt worden, als een soort uitgebreide batch of samen met HTML. Het is een open-source taal welke gratis kan gebruikt worden. Een script werkt als volgt. Een browser opent een PHP pagina zoals hij een HTML pagina opent. De server verwerkt dit script en geeft als resultaat een pagina weer opgesteld in HTML. De HTML-tags moeten door de gebruiker wel expliciet in het script meegegeven worden. Men moet er op letten dat PHP bij de verwerking verschilt van sommige andere scripttalen. Zo wordt een PHP-script lijn per lijn verwerkt en ook zo weergegeven naar de browser toe. Dit wil zeggen van zodra het script iets uitschrijft dit weergegeven wordt in de browser en niet herroepen kan worden. De pagina wordt dus niet eerst op de server in één geheel opgesteld en pas daarna verzonden. Dit kan handig zijn om weten voor bijvoorbeeld foutmeldingen. Dit is volgens sommigen een

van de tekortkomingen in PHP. Daarnaast worden er nog vele tekortkomingen verweten, meer bepaald veiligheidsrisico's en het niet standaard aanwezig zijn van vele functies. Het eerste kan verholpen worden door goed rekening te houden met veiligheid tijdens het programmeren. Het tweede geeft voornamelijk problemen als de programmeur geen eigenaar is van de server en dus geen toegang heeft tot de instellingen van de PHP-engine. Dit kan soms verholpen worden door in het script de functies expliciet te initialiseren of de constanten in te stellen, maar vaak heeft men daar geen controle over. Dit is dus een tekortkoming waar moet mee geleefd worden, door bijvoorbeeld de functies zelf te implementeren.

3.1.2 Syntax en gebruik

De code in een script begint altijd met `<?php` en eindigt met `?>`. Deze kan eender waar in het script voorkomen en zelfs meerdere malen. Dit betekent dus dat PHP-code verweven kan worden met HTML, enkel hetgeen tussen `<?php` en `?>` staat wordt door de engine aanzien als code, de overige tekens als platte tekst. Ieder commando of functie wordt afgesloten met een puntkomma `;` zoals in vele andere programmeertalen. Variabelen worden voorafgegaan door een dollarteken `$` zo is bijvoorbeeld `$var` een variabele. Toekenning van waarden aan variabelen gebeurt door het gelijkheidsteken `=`. De variabele `$var` krijgt de waarde 5 via `$var = 5;`. Ook kunnen de waarden van variabelen opgegeven worden door functies, `$var = strtoupper("kleine letters worden hoofdletters")`. Men kan een pagina (script) of een reeks van pagina's met één bepaalde gebruiker associëren door gebruik te maken van sessions. Bij deze wordt er op de server een bestand aangemaakt waarmee het script gegevens kan uitwisselen. Dit bestand wordt uniek aan een cookie gelinkt, welke door de gebruiker bijgehouden wordt. In volgende paragraaf wordt daar dieper op ingegaan bij het bespreken van de verschillende functies.

3.1.3 Vaak gebruikte functies

Sessions

Sessions worden gebruikt om gegevens te personaliseren aan een bepaalde gebruiker en om gegevens tussen pagina's onderling uit te wisselen. Als men sessions wil gebruiken moet deze op iedere pagina eerst geïnitieerd worden door het plaatsen van het commando

```
session_start(); (3.1)
```

in het begin. Daarna kan men eenvoudig gegevens uitwisselen via

```
$_SESSION["variabele"] = waarde; (3.2)
```

hierbij is *variabele* de naam van de variabele in het sessionbestand en *waarde*, de waarde die variabele krijgt. Om gegevens uit het sessionbestand te halen gebruikt men opnieuw bovenstaande functie met dat verschil dat de waarde aan een variabele toegewezen wordt.

$$\$var = \$_SESSION["variabele"]; \quad (3.3)$$

Als men een sessiebestand wil vernietigen dan kan dat expliciet gedaan worden via de functie

$$session_destroy(); \quad (3.4)$$

Indien dit niet in de code vermeld wordt zal een sessie sowieso door de server na een vooraf bepaalde tijd vernietigd worden, meestal is dat 20 minuten.

Bewerkingen op variabelen

Het toewijzen van een waarde is reeds besproken. De eenvoudigste functies zijn de elementaire wiskundige bewerkingen.

$$\begin{aligned} \$x &= 5; \\ \$y &= \$x + 5; \end{aligned}$$

Dit geeft voor *y* het resultaat 10. De vergelijkings- en logische operatoren volgen eenzelfde stramien. Om te weten of een bepaalde variabele geïnitieerd is gebruikt men

$$isset(variabele); \quad (3.5)$$

Dit geeft TRUE indien variabele geïnitieerd is en FALSE indien niet. Het omgekeerde van de *isset*-functie vernietigt een variabele.

$$unset(variabele); \quad (3.6)$$

Controlestructuren

Controlestructuren zijn functies die controleren of aan een bepaalde voorwaarde voldaan is. Zoja, dan wordt een stuk code uitgevoerd. Er is vooreerst de gekende if-then-else structuur.

```

if(uitdrukking){
    code indien aan de uitdrukking voldaan is
}
else{
    code indien niet aan de uitdrukking voldaan is
}

```

(3.7)

Een gelijkaardige functie is de switch. Bij deze wordt een variabele vergeleken met verschillende mogelijke waarden. Indien de variabele met een bepaald geval overeenkomt wordt de bijpassende code uitgevoerd en alle daaronder staande code, behalve als er een break staat, dan wordt enkel de code die bij de desbetreffende case hoort uitgevoerd.

```

switch$var{
    casewaarde1 :
        code
        break;
    casewaarde2 :
        code
        break;
}

```

(3.8)

Daarnaast bestaan er nog iteratieve lussen, zoals de while-lus. Zolang er aan de uitdrukking voldaan is zal de code uitgevoerd worden.

```

while(uitdrukking){
    code
}

```

(3.9)

De meest gekende iteratieve lus is de for-lus.

```

for(uitdrukking1; uitdrukking2; uitdrukking3){
    code
}

```

(3.10)

Hierbij wordt bij het begin van de for-lus *uitdrukking1* uitgevoerd. Bij het begin van iedere iteratie wordt *uitdrukking2* gevalueerd. Indien deze waar is, wordt de code uitgevoerd. Op

het einde van iedere iteratie wordt dan uitdrukking3 uitgevoerd.

Werken met bestanden

Scripts zijn voornamelijk ontworpen om repetitieve taken uit te voeren op bestanden. Voornamelijk tekstbestanden. Er bestaat dan ook een heel gamma aan functies die inwerken op bestanden en strings. In de volgende paragraaf wordt er wat dieper ingegaan op het werken met strings. Voor er met bestanden kan gewerkt worden dient men deze eerst te openen. Dit gebeurt door een bestandsnaam aan een stream te koppelen. Dit is een soort link waardoor het script weet naar welke file er verwezen wordt. Zo kan men op een bepaald moment verschillende bestanden geopend hebben en die simultaan gebruiken. Het koppelen van een stream aan een bepaald bestand gebeurt met volgend commando

$$\$stream = fopen(bestandsnaam, mode); \quad (3.11)$$

Hierbij is bestandsnaam het pad naar de bestandsnaam en mode wat de stream kan, \$stream is de link naar het bestand, dit noemt men een resource handler. De mode r staat voor alleen lezen, w voor alleen schrijven en r+ voor lezen en schrijven. Om te weten waar in het bestand men op een bepaald moment zit, werkt het systeem intern met file pointers. Bij de vorige modes wordt bij het openen van de stream deze pointers aan het begin van het bestand geplaatst. Van zodra er een bepaalde handeling gebeurt, zal de pointer veranderen. Bijvoorbeeld na het lezen van het eerste karakter zal de pointer één plaats opgeschoven zijn. Naast de zojuist vermelde modes bestaan er nog een aantal andere, meer bepaald voor het aanmaken van een bestand en om de wijzer aan het eind in plaats van het begin van een bestand te plaatsen. Eenmaal geopend kan men verschillende bewerkingen uitvoeren. Het lezen gebeurt met volgende functie

$$fgetc(\$stream). \quad (3.12)$$

Deze functie leest het karakter dat op de plaats van de pointer staat en schuift de pointer een positie op. Om een hele lijn in te lezen maakt men gebruik van

$$fgets(\$stream) \quad (3.13)$$

Waar toe 3.12 en 3.13 dienen kan men makkelijk herkennen aan het laatste karakter van de functie, c staat voor character en s voor string. De functie waarvan, in deze thesis, het meest gebruik van wordt gemaakt bij het inlezen van bestanden is

$$fgetcsv(\$stream, lengte, separatieteken). \quad (3.14)$$

Deze leest een hele lijn van een CSV bestand in en plaatst de strings, afgescheiden door een separatieteken, in een array. De lengte geeft weer hoeveel karakters van het bestand moet ingelezen worden. Het is sterk aangeraden deze groter te nemen dan een lijn in het CSV bestand lang is. Het separatieteken is het teken waarmee in het CSV bestand de verschillende velden van elkaar afgescheiden worden. In het volgend hoofdstuk gaat men dieper op in wat een CSV is. Wil men schrijven naar een bestand, dan gebeurt dit via

$$fwrite(\$stream, string) \quad (3.15)$$

Wanneer het nodig is te weten of men het einde van een bestand bereikt heeft, gebruikt men volgende booleaanse functie

$$feof(\$stream). \quad (3.16)$$

Deze geeft TRUE indien de pointer het einde van een bestand bereikt heeft en FALSE indien dit niet zo is. Nadat men klaar is met een bestand, sluit men de stream daarvoor af via

$$fclose(\$stream). \quad (3.17)$$

Het kopiëren van een bestand doet men via

$$copy(bron, doel). \quad (3.18)$$

Een iets apart staand geval is de variabele FILES. Deze wordt gebruikt bij het uploaden van bestanden via het HTTP. Via deze kan men het pad van het upgelode bestand te weten komen

$$\$_FILES["naam"]["tmp_name"] \quad (3.19)$$

Hierbij is naam de naam die meegegeven is aan het bestand bij het uploaden. "tmp_name" moet altijd aanwezig zijn. Indien men andere informatie te weten wenst te komen over het upgelode bestand, zoals bijvoorbeeld hoe groot het is, dan moet men tmp_name vervangen door een ander commando. Aangezien dit niet gebruikt wordt in deze thesis gaan we er niet dieper op in. Naast de zojuist kort besproken functies voor het behandelen van bestanden bestaan er nog tal van andere, maar daarvan wordt er geen gebruik gemaakt bij het aanmaken, beheren en gebruiken van het evaluatiesysteem.

Het werken met strings en arrays

Arrays zijn rijen van strings of andere objecten. Een string is een rij karakters. Een array wordt aangemaakt via

$$\$arraynaam = array(object1, object2, \dots); . \quad (3.20)$$

Wil men weergeven wat er op een bepaalde plaats i in de array staat, dan wordt dit gedaan als volgt

$$\$arraynaam[i]; \quad (3.21)$$

Waarbij i staat voor een getal $0,1,2,3,\dots$. Om een array te maken die bestaat uit een opeenvolging van elementen, bijvoorbeeld letters uit het alfabet, dan maakt men gebruik van

$$range('begin', 'eind') \quad (3.22)$$

waarbij *begin* staat voor het begin van de opeenvolgende elementen en *eind* waar de array moet stoppen. Zo geeft `range('a','z')` de array `(a,b,c,\dots,x,y,z)`. Om verschillende rijen aan elkaar te koppelen tot een grote array, moet men deze mergen

$$array_merge(array1, array2, \dots) \quad (3.23)$$

Voor bewerkingen op strings bekijken we drie functies. Het verwijderen van deelstrings uit een string wordt gedaan door de functie

$$str_replace(zoekstring, vervangstring, string) \quad (3.24)$$

Hierbij is *zoekstring* de string die dient vervangen te worden, *vervangstring*, de string die in de plaats van *zoekstring* komt en *string*, de string waarin gezocht wordt. Wil men van een bepaalde string slechts een deel hebben dan gebruikt men

$$substr(string, positie1, positie2) \quad (3.25)$$

Hier wordt er een deelstring weergegeven van string vanaf positie1 tot positie2. Tot slot kan men een string versleutelen volgens de md5 methode via

$$md5(string) \quad (3.26)$$

PHP en MySQL

Voor de interactie van PHP met de databasetaal MySQL bestaan er ook functies. Het connecteren naar een database doet men via

$$mysql_connect('localhost', gebruiker, wachtwoord) \quad (3.27)$$

Hierbij wordt een connectie gemaakt met MySQL dat zich bevindt op localhost met als gebruikersnaam gebruiker en wachtwoord wachtwoord. Eenmaal men een connectie heeft moet men een database selecteren

$$mysql_selectdb(naambd, connect) \quad (3.28)$$

Hierbij selecteert men de database maandb en dit met connectie connect, dit is de handler geopend via mysql_connect. Als men een MySQL query wil uitvoeren dan gebruikt men

$$mysql_query(opdracht) \quad (3.29)$$

Dit voert de MySQL opdracht uit in de databank. Wil men de resultaten van een query weergeven dat gebeurt dit via

$$mysql_result(query, i) \quad (3.30)$$

Een query geeft altijd een array weer als resultaat, wil men weten wat er op plaats i in de array staat dan gebruikt men 3.30. Als men klaar is met het gebruik van de database sluit men de connectie af

$$mysql_close(connect). \quad (3.31)$$

Andere functies

Tot slot bespreken we de andere gebruikte functies die niet in een categorie ingedeeld worden. Een mail versturen doe je als volgt

$$\text{mail}(\text{aan}, \text{onderwerp}, \text{bericht}). \quad (3.32)$$

Fouten worden opgevangen door de functie

$$\text{die}(\text{string}) \quad (3.33)$$

Wanneer er iets fout loopt en die staat er als foutopvang dan wordt het script onmiddellijk verlaten en wordt string als boodschap meegegeven. Om iets uit te schrijven naar het scherm bestaat er een simpel commando dat een string weergeeft

$$\text{echo}(\text{string}). \quad (3.34)$$

Wil men de datum weergeven in een zeker formaat, bijvoorbeeld d voor de dag, m voor de maand en Y voor het jaar,

$$\text{date}(\text{formaat}) \quad (3.35)$$

Als men met een form-formulier werkt en men wil de data daaruit verkrijgen dan wordt dit gedaan via

$$\$_GET['naam'] \quad (3.36)$$
$$\$_POST['naam'] \quad (3.37)$$

Waarbij naam staat voor de naam die de variabele kreeg in het formulier. Om een HTTP header te verzenden

$$\text{header}(\text{string}, \text{false}) \quad (3.38)$$

Dit verzendt de header met string. False is optioneel, indien dit vermeld staat, wil dit zeggen dat als er iets fout loopt deze header verzonden wordt.

3.2 De database MySQL

Eerst wordt kort toegelicht wat MySQL is, daarna worden enkele commando's besproken.

3.2.1 Theoretische achtergrond

MySQL is een databanktaal gebaseerd op de SQL standaard. Deze is gratis in gebruik onder de GPL licentie. Een databank kan gezien worden als een verzameling van gegevens. Deze worden geordend in tabellen. De standaard voor databanken is SQL ontwikkeld in 1986. Sindsdien zit men al aan versie SQL2003, wat een lichte uitbreiding is van SQL3. MySQL is gebaseerd op SQL, doch doet er niet volledig aan. De laatste versies trachten dit in te halen, maar er ontbreken nog belangrijke functionaliteiten. Zo is het bijvoorbeeld niet mogelijk om gebruik te maken van vreemde sleutels, is de beveiliging bij falen niet hoog (als bijvoorbeeld de stroom uitvalt kunnen nog niet opgeslagen gegevens verloren gaan). Er is dus veel kritiek op MySQL en velen begrijpen niet waarom andere open source alternatieven die "beter" zijn niet worden gebruikt. De reden hiervoor is simpel, het is eenvoudig te installeren en voor basisgebruik ook heel eenvoudig. Het is het meest verspreide systeem bij providers die een gratis databankbeheersysteem aanbieden, dit wellicht aangezien het makkelijk is in onderhoud. De provider waarop het evaluatiesysteem zal draaien maakt gebruik van MySQL en dit systeem voldoet geheel aan onze behoeften. Dit zijn de twee redenen waarom we als databanksysteem MySQL nemen. In volgende paragraaf bespreken we de verschillende uitdrukkingen, edoch vermelden we hier in welke volgorde die door het systeem uitgevoerd worden. Eerst wordt de FROM clause bekeken. Daarna evalueert het systeem de WHERE clause en vervolgens de SELECT clause. Het eindigt bij de ORDER BY clause. Waarna het resultaat weergegeven wordt. Het valt op dat er in hoofdletters geschreven wordt. Dit is niet verplicht, het systeem kan de functies ook in kleine letters lezen. Maar bij conventie worden alle gekende uitdrukkingen in hoofdletters geschreven. In wat volgt worden alle uitdrukkingen die door ons gebruikt worden toegelicht.

3.2.2 De verschillende MySQL uitdrukkingen

SELECT FROM

De meest elementaire uitdrukking is de SELECT FROM. Deze bestaat uit twee delen. De FROM, welke aanduidt uit welke tabellen uit de database men iets wil selecteren. En de SELECT, welke aanduidt welke attributen men wil selecteren. Deze attributen worden gescheiden door komma's. Indien men alle attributen wil dan plaats men een asterisk *

```
SELECT attribuut1, attribuut2,... FROM tabel (3.39)
```

WHERE

De WHERE clause is een conditionele selectie. Deze geeft de voorwaarden waaraan de attributen moeten voldoen. De WHERE clause komt na de FROM.

```
SELECT ... FROM ... WHERE conditie (3.40)
```

De SELECT wordt uitgevoerd als aan de conditie voldaan is. Deze kan bestaan uit vergelijking, een logische uitdrukking, of een bepaald veld leeg is (via IS NULL),...

JOIN operatoren

De JOIN is het cartesisch product van twee tabellen met een voorwaarde. Het gewoon cartesisch product wordt gegeven door de CROSS JOIN

```
SELECT * FROM A CROSS JOIN B (3.41)
```

Hoe dit eruit ziet wordt weergegeven in figuur 3.1. Naast de CROSS JOIN bestaat de INNER JOIN. Dit is een JOIN waar er gebruik wordt gemaakt van een bepaald attribuut. Dit wordt met beide tabellen vergeleken, en de tuples waarvan het veld in beide tabellen voorkomt wordt weergegeven.

```
SELECT * FROM A INNER JOIN B USING(Y) (3.42)
```

Hoe dit eruitziet wordt weergegeven in figuur 3.2.

INSERT

Na het bespreken van de bewerkingen op tabellen, wordt nu nagegaan hoe de structuren aangemaakt kunnen worden. We beginnen met het invoeren van gegevens in tabellen. Dit gaat via het commando INSERT

```
INSERT INTO tabel (attribuut1, attribuut2, ...) VALUES ('waarde1','waarde2',... (3.43)
```

Hier worden de attributen attribuut1, attribuut2,... in de tabel geplaatst met als respectievelijke waarden waarde1, waarde2, ... In deze waarden kunnen andere MySQL functies staan, zoals een SELECT clause.

Tabel A					
X	Y	A.X	A.Y	B.Y	B.Z
A1	B1	A1	B1	B1	C1
A2	B2	A1	B1	B2	C2
A3	B3	A1	B1	B4	C4
		A2	B2	B1	C1
		A2	B2	B2	C2
		A2	B2	B4	C4
		A3	B3	B1	C1
		A3	B3	B2	C2
		A3	B3	B4	C4

Figuur 3.1: Het cartesisch product van de tabellen A en B.

Creëren van een database

Natuurlijk zijn we niets met bovenstaande commando's als we geen databank hebben. Dit gebeurt via het eenvoudige commando

```
CREATE DATABASE naam (3.44)
```

Hierbij is naam de naam van de database.

Aanmaken van tabellen

Het aanmaken van tabellen volgt op het creëren van de database.

```
CREATE TABLE naam(at1 eigenschappen, at2 eigenschappen, PRIMARY KEY(at1) (3.45)
```

Dit maakt een tabel aan met naam naam, attributen at1 en at2 elk gevolgd door hun eigenschappen en met primaire sleutel at1. De eigenschappen zijn welk soort variabele het is, bijvoorbeeld een integer, of het veld NULL kan zijn of niet, ...

Tabel A	<table style="border-collapse: collapse; width: 100%;"> <tr> <th style="padding: 2px 10px;">X</th> <th style="padding: 2px 10px;">Y</th> </tr> <tr> <td style="padding: 2px 10px;">A1</td> <td style="padding: 2px 10px;">B1</td> </tr> <tr> <td style="padding: 2px 10px;">A2</td> <td style="padding: 2px 10px;">B2</td> </tr> <tr> <td style="padding: 2px 10px;">A3</td> <td style="padding: 2px 10px;">B3</td> </tr> </table>	X	Y	A1	B1	A2	B2	A3	B3
X	Y								
A1	B1								
A2	B2								
A3	B3								
Tabel B	<table style="border-collapse: collapse; width: 100%;"> <tr> <th style="padding: 2px 10px;">Y</th> <th style="padding: 2px 10px;">Z</th> </tr> <tr> <td style="padding: 2px 10px;">B1</td> <td style="padding: 2px 10px;">C1</td> </tr> <tr> <td style="padding: 2px 10px;">B2</td> <td style="padding: 2px 10px;">C2</td> </tr> <tr> <td style="padding: 2px 10px;">B4</td> <td style="padding: 2px 10px;">C4</td> </tr> </table>	Y	Z	B1	C1	B2	C2	B4	C4
Y	Z								
B1	C1								
B2	C2								
B4	C4								

SELECT *
FROM A INNER JOIN B USING (Y)

A.X	A.Y	B.Y	B.Z
A1	B1	B1	C1
A1	B1	B2	C2
A1	B1	B4	C4
A2	B2	B1	C1
A2	B2	B2	C2
A2	B2	B4	C4
A3	B3	B1	C1
A3	B3	B2	C2
A3	B3	B4	C4

Figuur 3.2: De INNER JOIN van de tabellen A en B via het kolom Y.

3.2.3 Gevaarlijk probleem bij gebruik PHP en MySQL

Er is reeds aangehaald dat er problemen zijn met zowel PHP als MySQL betreffende de veiligheid. Het probleem waar we mee te maken hebben is het inloggen. Dit wordt gedaan door de gebruiker een username te laten ingeven. Gevaar bestaat dat iemand met minder goede bedoelingen misbruik hiervan kan maken door ipv een naam een MySQL string in te geven. Meer bepaald volgende

DROP TABLE users; SELECT * FROM data WHERE name LIKE '%' (3.46)

Dit zal dan bijvoorbeeld in SELECT terecht komen

SELECT * FROM users WHERE name = 'a'; DROP TABLE users;
SELECT * FROM data WHERE name LIKE '%'; (3.47)

Dit noemt men SQL injection. Dit delete de users tabel en geeft de inhoud van een andere tabel weer. Op die manier kan de users tabel aangepast worden en alle inhoud weergegeven. Ook kan men door van dergelijke systemen gebruik te maken inloggen, ook al kent men het wachtwoord en de username niet. Met dergelijke zaken dient men rekening te houden.

3.3 De LaTeX markuptaal

LaTeX is een beschrijvingstaal om documenten op te maken. Deze maakt gebruik van TeX. Ze wordt vooral in de drukkers- en wetenschappelijke wereld gebruikt. Ze is ideaal om wiskundige en chemische formules weer te geven. Een beschrijvingstaal wil zeggen dat LaTeX beschrijft hoe uw pagina er uit moet zien. Het zegt niet waar op uw blad een titel moet komen, alhoewel je dat wel expliciet kan vermelden. De kracht zit hem dat LaTeX voor u uw blad schikt. Dit wordt op zo een manier gedaan, dat automatisch aan de standaard van een tekst voldaan is. Er wordt gezorgd dat er zo weinig mogelijk pagina's worden ingenomen, dat de tekst mooi over het blad is verdeeld, afbreking van woorden, aanmaken van inhoudstafels, referenties, bibliografieën,... wordt ook voor u gedaan. Kortom u hoeft u als gebruiker enkel te bekommeren om de inhoud van de tekst. Ook is LaTeX al jaar en dag een standaard die nog niet veranderd is en nog aan alle eisen van de hedendaagse gebruiker voldoet. Uw teksten in LaTeX typen zorgt er dus voor dat de kans heel groot is dat ze over 20 jaar nog leesbaar zijn. Natuurlijk u heeft met een .tex bestand geen direct leesbare tekst. Deze moet gecompileerd worden. Daarbij kan u kiezen naar welk formaat. De meest gebruikte zijn postscript (.ps) en PDF. Postscript is de taal van uw printer en PDF wordt wereldwijd gebruikt. Ook is tot nu toe PDF backwards compatibel. Dit wil zeggen dat u met een nieuwe PDF viewer uw oude documenten nog kan lezen. In wat volgt bespreken we kort hoe een document wordt opgebouwd en enkele tags.

3.3.1 Opbouw en tags

Een TeX document bestaat uit hoofdstukken, die op zich bestaan uit kleinere eenheden. Men kan kiezen de hoofdstukken in afzonderlijke bestanden onder te brengen of ze als 1 groot bestand te zien. Iedere LaTeX file heeft de extensie .tex. Welke standaard men gebruikt voor een document wordt weergegeven in preamble. Daarin worden ook andere eigenschappen geplaatst. Voor u als gebruiker maakt dit niet zoveel uit, dit is enkel van belang bij het compileren en op het net zijn er meer dan genoeg voorgemaakte preambles te vinden. Eenmaal u uw tekst af hebt dan compilet u. U moet dit een drietal keer achtereen doen, zodat de referenties goed zijn. Het meest kenmerkende in LaTeX is hoe de tags worden aangeduid. Deze worden voorafgegaan door middel van een backslash \. In wat nu volgt bespreken we enkele vaak voorkomende tags en degene die we gebruiken om de rapporten aan te maken.

Structuur

Een document bestaat meestal uit hoofdstukken die zelf bestaan uit secties, subsecties, paragrafen,...

$$\backslash\text{chapter}(\text{titel}) \tag{3.48}$$

$$\backslash\text{section}(\text{titel}) \quad (3.49)$$

$$\backslash\text{subsection}(\text{titel}) \quad (3.50)$$

Een pagina einde wordt weergegeven door

$$\backslash\text{newpage} \quad (3.51)$$

Tabellen

Tabellen worden in een tabular omgeving gemaakt.

$$\backslash\text{begin}\{\text{tabular}\}\{\text{positie}\}\{\text{kolommen}\} \text{rijen} \backslash\text{end}\{\text{tabular}\} \quad (3.52)$$

Rijen zijn de gegevens in de tabel. Kolommen zijn het aantal kolommen, daarin wordt ook aangegeven hoe de uitlijning is l voor links, r voor rechts en c voor gecentreerd. Rijen worden van elkaar gescheiden door $\backslash \backslash$. In een rij worden de verschillende kolommen van elkaar gescheiden door $\&$. Een horizontale lijn wordt geplaatst door

$$\backslash\text{hline} \quad (3.53)$$

Een verticale lijn door

$$\backslash\text{vline} \quad (3.54)$$

Tekst

Tekst kan men vet, schuin, groot, klein, ... maken. Daar bestaan ook commando's voor. Vette tekst door

$$\backslash\text{textbf}\{\text{tekst}\} \quad (3.55)$$

Schuine tekst door

$$\backslash\text{textit}\{\text{tekst}\} \quad (3.56)$$

Grote en kleine tekst door

`\large{tekst}` (3.57)

`\small{tekst}` (3.58)

Het centreren door

`\begin{center}` te centreren tekst `\end{center}` (3.59)

Dit zijn de meest courante commando's voor het opmaken van een rapport in LaTeX.

Hoofdstuk 4

Bespreking van de oplossing op technisch niveau

4.1 Ontwerp van de databank

4.1.1 Toelichting bij de ontwerpmethodologie

Zoals reeds vermeld werd in hoofdstuk 1, was het al snel duidelijk dat er gewerkt zou worden met een *relationeel* databankmodel. Volgens dit model worden alle gegevens in een databank gezien als tabellen en niets anders dan tabellen. Een tabel is een voorstellingsvorm van het abstract wiskundig begrip ‘relatie’. Een relatie is op zijn beurt een verzameling van geordende n -tupels. Er diende dus eerst bepaald te worden hoeveel relaties/tabellen er nodig waren. Aan elke relatie moesten vervolgens een aantal (n) attributen gekoppeld worden (deze vormen de ‘intentie’ van de relatie). Eenmaal de relaties en de attributen gedefinieerd waren, zouden de tabellen dan opgevuld kunnen worden met de data (deze data-tupels vormen dan de ‘extentie’ van de relatie). Absolute voorwaarde bij het opvullen van tabellen van een relationele databank is dat er geen dubbele rijen (= data-tupels) aanwezig zijn. Daarnaast dienen de ingevulde attribuutwaarden (de waarde voor een cel op de kruising van een bepaalde rij en kolom) steeds scalair te zijn (niet verder opsplitsbaar). Bij het implementeren van de relaties in MySQL moeten ook de domeinen van de attributen vastgelegd worden. Concreet betekent dit dat men het datatype van elke kolom moet instellen. In MySQL zal ook de collatie (tekenset) van de attributen bepaald moeten worden.

Om dus (het aantal) relaties en attributen te bepalen, werd eerst een ‘Entity/Relationship’-model opgesteld. Een dergelijk model wordt vaak als tussenstap gebruikt bij het ontwerp van databanken en is *niet* bedoeld om rechtstreeks geïmplementeerd te worden. In de plaats daarvan kan een dergelijk diagram wel omgezet worden naar een databankschema (hier relationeel). Vaak wordt gestart van een beschrijvende tekst. Daaruit worden entiteiten (dit zijn fysische of conceptuele zaken die een zelfstandig bestaan leiden in de echte wereld) en

attributen (eigenschappen van de entiteiten) afgeleid. Er bestaan verschillende entiteitstypes; daarnaast zijn er ook zgn. relatietypes (die de relatie tussen entiteiten bepalen). Relatietypes, entiteiten en attributen worden via gestandaardiseerde symbolen voorgesteld, resp. ruiten, rechthoeken en ovaal. Ook de kardinaliteit van elk relatietype (hoeveel entiteiten zijn er op een gegeven ogenblik maximaal betrokken in een relatie? aangegeven door bv. 1:N of M:N) en de participatiegraad (is op elk ogenblik elke entiteit van een bepaald entiteitstype betrokken in een relatie van het relatietype? aangegeven door enkele of dubbele lijnen) worden in een dergelijk diagram aangeduid.

Waarmee diende er rekening gehouden te worden tijdens het ontwerp van de databank en welke veronderstellingen werden er gemaakt?

- allereerst werd besloten om één databank (= set van tabellen) aan te maken per schooljaar. Dit systeem werd verkozen boven het toevoegen van een attribuut ‘jaar’ aan de tabellen, wat de zaken te complex zou maken. Voor het bevragen van de gegevens van vorige schooljaren hoefde dit geen belemmering te vormen, en met het oog op het nemen van database-dumps was het eveneens beter om per jaar reservekopieën te maken i.p.v. telkens alle historische data mee te ‘dumpen’.
- hoewel de databank in de eerste plaats de evaluatie van leerlingen dient te ondersteunen, moeten er ook administratieve gegevens in de databank gestockeerd kunnen worden. Het is niet de bedoeling dat deze databank gebruikt wordt voor administratieve doeleinden; deze gegevens (adres, geboortedatum, telefoon, geslacht, geboortedatum,...) zijn echter nodig voor de rapportering (vermelding van bepaalde zaken op het rapport) of om de leerkracht enige achtergrond te bieden bij de beoordeling (zo zal bv. de leeftijd berekend worden uitgaande van de geboortedatum).
- één van de vereisten (zie paragraaf 2.2) was dat de beoordelingscriteria op jaarlijkse basis per vak en per graad konden veranderd worden. Wat het type van de criteria betreft kan er onderscheid gemaakt worden tussen vijf beoordelingsvormen:
 - door middel van tekst
 - door middel van een cijfer op een zelf te bepalen maximum
 - door middel van een niveau (gebruikt in de sectie muziek), waarbij men op het rapport iets dergelijks aantreft:
Niv.1 || **Niv.2** || Niv.3 || Niv.4 || Niv.5 || Niv.6 || Niv.- of n.v.t.
 - door middel van een code (gebruikt in de sectie woord), waarbij men analoog iets van deze aard krijgt:
a || b || **c** || d || e of n.v.t.
 - door middel van een gradatie (men kiest een optie uit een keuzelijst: ‘zeer goed’, ‘goed-zeer goed’, ‘goed’, ...)

Voor die vakken die daarenboven een partieel examen rond Pasen organiseren, heeft men bijkomend de volgende mogelijkheden:

- een globaal cijfer op een zelf te bepalen totaal
- opdeling in een aantal rubriekjes, die afzonderlijk gequoteerd worden en die samen het behaalde totaal geven

Om het systeem zo flexibel mogelijk te maken, werd ervoor gekozen om deze opties i.v.m. de criteria en de examens voor alle vakken ter beschikking te stellen. Daar waar men in het oorspronkelijke systeem nog de toegelaten opties per vak moest (hard)coderen, zal men nu zelf per vak (en per trimester, althans voor de examens) kunnen vastleggen voor welke manier van werken men opteert. In het verdere verloop van paragraaf 4.1 zal dit nog duidelijker worden.

- verder werd er uitgegaan van het feit dat er steeds een unieke koppeling bestaat tussen een klasgroep en een bepaalde docent. Zoals uit het Entity/Relationship-diagram zal blijken, wordt een klasgroep uniek bepaald door de graad, de optie en het vak die de leerlingen uit die klasgroep allen volgen, in combinatie met een ‘identifieer’ voor de klasgroep zelf.
- de andere premisses worden in de beschrijvende tekst verderop aangehaald.

4.1.2 Opstellen van het relationeel databankmodel

Beschrijvende tekst

De volgende descriptieve tekst werd opgesteld na het aanhoren van de verwachtingen betreffende het nieuwe systeem en na het bestuderen van de organisatiestructuur van de academie (zie Figuur 4.1). De entiteiten werden aangeduid met blokjes, de relatietypes (die eerder het verband tussen entiteiten aangeven) werden in het cursief geplaatst. De attributen die bij een bepaalde entiteit horen, kunnen afgeleid worden uit de tekst (hier niet aangeduid om het overzicht te bewaren).

De Stedelijke Academie van Deinze organiseert cursussen in 3 studierichtingen:

- muziek
- woord
- dans

Deze academie heeft behalve in Deinze ook afdelingen in Kruishoutem, Nazareth, Sint-Martens-Latem en Zulte.

Cursisten geven bij *inschrijving* een aantal persoonlijke gegevens op nl. hun naam en voornaam, geslacht, adres (straat, nummer, bus, postcode, gemeente, land), geboortedatum en één of twee telefoonnummers (tel./GSM). Daarbij wordt aan elke cursist een uniek 4-cijferig stamnummer toegekend.

Van de **leerkrachten** houdt men naam en voornaam bij, hun persoonlijk e-mailadres evenals de sectie (muziek, woord of dans) waarin ze les geven. Ze worden uniek gekarakteriseerd door hun werknemernummer. Leerkrachten dienen tevens een gebruikersnaam en wachtwoord (dit laatste wordt geëncrypteerd) op te geven om in te loggen op het centraal evaluatie- en rapporteringssysteem. Voor directie- en personeelsleden die eveneens van dit systeem gebruik wensen te maken, worden — met uitzondering van de sectie — identiek dezelfde gegevens geregistreerd.

Cursisten kiezen voor één of meerdere **opties** (cfr. lichtgrijze kadertjes in Figuur 4.1) binnen de lagere, middelbare of hogere graad. Hun progressie binnen die **graad** wordt aangegeven door het jaar waarin ze zich bevinden (hiervoor wordt een code gebruikt; bv. L4, M2, H3, ...). De keuze voor een bepaalde optie impliceert dat de cursist een aantal verplichte **vakken** en/of keuzevakken (gekarakteriseerd door een naam en een vaknummer) dient te volgen. Het volgen van een optie in een hogere graad (M of H) is enkel mogelijk na het succesvol vervolmaken van bepaalde opties in de daaraan voorafgaande (lagere) graden (voorkennis is dus vereist). Het toelaten van cursisten in deze hogere graden staat onder de controle van de leerkrachten en van het secretariaat. Een cursist kan tijdens een lopend schooljaar echter ook voor opties in verschillende graden (zelfs van verschillende studierichtingen) ingeschreven zijn — bv. wanneer een student klarinet in H1 beslist ook de cursus hobo in L2 te gaan volgen — en kan dan eventueel vrijgesteld worden voor enkele van de verplichte vakken die aan die optie(s) verbonden zijn. Een vrijstelling van 50% betekent dat de cursist het bewuste vak op een andere academie volgt. Net als bij 100% vrijstelling houdt dit in dat de cursist voor het vak in kwestie niet beoordeeld wordt op de AMWD Deinze. In sommige gevallen kan het gebeuren dat een student tweemaal hetzelfde vak volgt tijdens hetzelfde jaar, maar bij verschillende leerkrachten (omdat men bv. instrumentaal ensemble kiest als verplicht vak én als keuzevak).

Er kunnen meerdere *lesgevers* zijn voor een bepaald vak in een bepaalde graad. Een vak kan ook deel uitmaken van verschillende opties. Niet in elke afdeling worden alle vakken gedoceerd. Leerkrachten kunnen hun leerlingen per vak opsplitsen in een aantal **groepen**. Groepen zijn bijgevolg uniek gelinkt aan een bepaalde leerkracht en bevinden zich onderaan in de hiërarchie 'graad-optie-vak-groep'. Grote groepen bevatten meestal leerlingen van dezelfde graad; kleine groepen zijn vaak samengesteld

uit leerlingen van meerdere graden. Leerlingen uit beide types groepen hoeven niet per se dezelfde optie te volgen. Drie maal per jaar (Kerst/Pasen/einde van jaar) worden de prestaties van de cursisten beoordeeld en wordt een rapport opgesteld. Dit zijn de zgn. 'evaluatiemomenten'. De (partiële) examens situeren zich rond Pasen en op het einde van het jaar. Niet alle vakken organiseren partiële examens. Leerkrachten hebben de mogelijkheid om zelf aandachtspunten of *criteria* te definiëren waarop hun cursisten zullen geëvalueerd worden (in onderlinge afspraak met die leerkrachten die hetzelfde vak geven in dezelfde graad). Bij het tweede en derde evaluatiemoment krijgen de leerlingen in de studierichting 'muziek' normalerwijze een cijfervak op 100 voor het partieel examen. De studierichting 'woord' gebruikt tot op vandaag nog geen cijfers; de richting 'dans' moet nog in het huidige systeem instappen.

Entity Relationship-diagram

Op basis van deze beschrijvende tekst kan nu een ER-diagram opgesteld worden. Hiervoor bestaat geen nauwkeurig beschreven procedure, het komt erop aan van het gezond verstand te gebruiken. Wel dienen de standaardsymbolen in hun correcte betekenis gebruikt te worden en moeten de sleutels (m.n. de identifiers die een entiteit uniek karakteriseren) in het diagram aangeduid worden. Betreffende de *participatiegraad* (enkele of dubbele lijnen voor resp. partiële en totale participatie) werden de volgende aannames gemaakt:

- een klasgroep dient altijd gekoppeld te zijn aan een leerkracht en anderzijds moet bij het toevoegen van een leerkracht aan de tabel deze ook direct verbonden worden met een klasgroep (beide entiteiten participeren dus totaal in het relatietype 'docent').
- een cursist moet bij inschrijving altijd gekoppeld worden aan een klasgroep, maar een klasgroep hoeft niet per se op het moment van de definitie cursisten te bevatten. Wel dient een klasgroep altijd toe te behoren aan een bepaald vak; bij de definitie van een vak hoeft het vak nog niet gelinkt te zijn aan klasgroepen (bv. omdat nog niet geweten is hoeveel leerlingen er zullen zijn en of opsplitsing in meerdere klassen nodig is).
- opties dienen altijd overeen te komen met één of meerdere vakken (volgt uit de beschrijvende tekst) en daaruit voortvloeiend dienen vakken dus ook aan een bepaalde optie gelinkt te zijn.
- tot slot dienen opties steeds toe te behoren aan een bepaalde graad, maar hoeven graden bij definitie nog niet gekoppeld te zijn aan bepaalde opties.

Men kan eventueel over deze aannames discussiëren, maar uiteindelijk heeft de participatiegraad weinig invloed op de uiteindelijke tabellenstructuur. Dit kan niet gezegd worden van de *kardinaliteitsratio* (mogelijkheden: 1:1, 1:N, N:M); deze verhouding speelt een belangrijke rol in het 7-delig algoritme dat gebruikt wordt om het ER-diagram om te zetten naar een relationeel schema. De interpretatie hieromtrent is als volgt:

- een cursist kan in meerdere klasgroepen zitten; een klasgroep bevat normaalgezien meerdere studenten.
- een leerkracht kan les geven aan meerdere klasgroepen (bv. opsplitsing in klas A, B en C bij een groot aantal leerlingen); een klasgroep is daarentegen uniek gelinkt aan één welbepaalde leerkracht
- door de hiërarchische opdeling (die te zien is in Figuur 4.1) komt een graad overeen met meerdere opties, een optie met meerdere vakken en een vak met meerdere klasgroepen. In de omgekeerde richting (zie Figuur 4.2) dient omwille van de identificerende relaties (zie verder) een klasgroep gekoppeld te zijn aan één vak, een vak aan één optie en een optie aan één graad.

Aan elke rechthoek (entiteit) in het ER-diagram (Figuur 4.2) werden de attributen toegevoegd (ovalen). Om de entiteiten uniek te kunnen identificeren binnen het relationeel model, moet telkens een attribuut van die entiteit als sleutel worden aangeduid. Deze sleutelattributen zijn te herkennen aan de onderlijnde vette tekst. De tabel ‘klasgroep’ is bijzonder omdat het hier een zwak entiteitstype betreft. Zwakke entiteiten (aangegeven door de dubbel omliggende kadertjes) zijn voor hun unieke identificatie afhankelijk van andere entiteiten, dit via zogenaamde identificerende relaties (dubbel omliggende ruiten). Concreet betekent dit dat een klasgroep — in tegenstelling tot bijvoorbeeld een cursist — enkel uniek geïdentificeerd kan worden door ook op de sleutels van één of meerdere andere entiteiten een beroep te doen. In de praktijk zal een klasgroep namelijk gekenmerkt worden door de *graadID*, *optieID* en *vakID* (= 3 vreemde sleutelattributen) en de *groepID* (= partieel sleutelattribuut, te herkennen aan gestippelde onderlijning) samen te nemen tot een primaire (partiële) sleutel (dit zijn samen 4 identifiers, dus 4 positieve gehele getallen). Analoog is ook elk vak afhankelijk van optie en graad (primaire sleutel uit drie IDs) en de optie afhankelijk via een identificerende relatie van de graad (2-delige sleutel).

De meeste enkelvoudige attributen spreken voor zichzelf. Aan de entiteit ‘cursist’ werd een samengesteld attribuut (‘adres’) toegevoegd. De meerwaardige attributen (dubbel omliggende ovalen) verdienen een woordje uitleg. Aan de entiteit ‘vak’ werd ‘criteria’ als een meerwaardig attribuut toegevoegd. Het verschil in interpretatie met het enkelvoudig attribuut ‘naam’ van diezelfde entiteit is dat — om bij dit concrete voorbeeld te blijven — een vak slechts één naam kan hebben, maar daarentegen meerdere beoordelingscriteria. Bij de conversie naar het relationeel schema zal een dergelijk meerwaardig attribuut aanleiding geven tot een aparte relatie, een van de tabel ‘vak’ gescheiden tabel ‘criteria’ met andere woorden. De attributen die in het schema aan ‘criteria’ werden toegekend, worden normaliter niet in het schema vermeld (evenals alle andere cursief geprinte attributen). Deze werden toegevoegd om de databankstructuur duidelijk af te beelden. Het nut van de ‘criteria’-attributen wordt later duidelijk, maar door deze manier van werken konden we de gevraagde variabiliteit en

flexibiliteit in de databank introduceren. Namelijk door ‘criteria’ als een meerwaardig i.p.v. een enkelvoudig attribuut te definiëren zal het mogelijk worden om elk jaar nieuwe criteria per vak en per graad vast te leggen (cfr. paragraaf 2.2).

Om deze criteria vervolgens te kunnen beoordelen, dienden tabellen toegevoegd te worden om de beoordelingen en cijfers in te stockeren. Aangezien de beoordeling afhankelijk is van de cursist en van de klasgroep (door de identificerende relatie kennen we meteen ook de graad, de optie en het vak) waarin de student les volgt, moeten deze tabellen tussen beide vernoemde entiteiten geplaatst worden, namelijk als (meerwaardige!) attributen van het relatietype ‘inschrijving’. Hadden we de beoordeling voor de diverse criteria als gewone attributen aan inschrijving toegevoegd, dan had dit als consequentie gehad dat het aantal criteria vastlag en dat er inefficiënt van de tabel ‘inschrijving’ gebruik zou moeten gemaakt worden zijn (immers, voor bepaalde vakken zouden de vooraf irreversibel gedefinieerde criteria niet van toepassing zijn, met half-lege kolommen tot gevolg). Er is in deze databankstructuur ook een logische en fysische scheiding tussen de beoordelingen van de drie evaluatiemomenten (dit zal de SQL-queries een stuk vereenvoudigen). Het attribuut ‘categorie’, dat zowel voorkomt bij de entiteit ‘criteria’ als bij ‘evaluatie_X’ — met $X = K(\text{erst}), P(\text{asen})$ of $E(\text{inde van jaar})$ — speelt een belangrijke rol in het toevoegen van bijkomende flexibiliteit aan de databank. Door dit attribuut kunnen criteria opgedeeld worden in categorieën (voor die vakken waar men dat wenst). Dit attribuut heeft echter een dubbele functie: in combinatie met het attribuut ‘trimester’ in de tabel ‘criteria’ kan meteen ook vastgelegd worden hoe de (partiële) examens voor een vak georganiseerd worden en van welk type deze examens zijn. Dit alles wordt in de verderop volgende voorbeelden duidelijk. Er werd ook één entiteit extra toegevoegd, de tabel ‘beheer’. Deze tabel moet los gezien worden van de andere tabellen en heeft als doel om statusinformatie (welk schooljaar is het? aan welk evaluatiemoment zijn we beland? wanneer ligt de deadline voor de rapporten?) door te geven van de beheerpagina’s (de beheerder voorziet in deze informatie) naar de gebruikerswebsite (gebruik in scripts of weergave op scherm).

De essentie van de zaak is dat we bij het ontwerp van de databank niet over één nacht ijs gegaan zijn en zoveel mogelijk ‘intelligentie’ trachten te steken hebben in de databank zelf. Door bepaalde attributen meerdere ‘taken’ te geven, konden we de databank compact houden (de 947 tabellen in de oorspronkelijke databank konden we reduceren tot 12+1 tabellen) ondanks de toegenomen mogelijkheden.

Relationeel schema, tabelstructuur en -eigenschappen

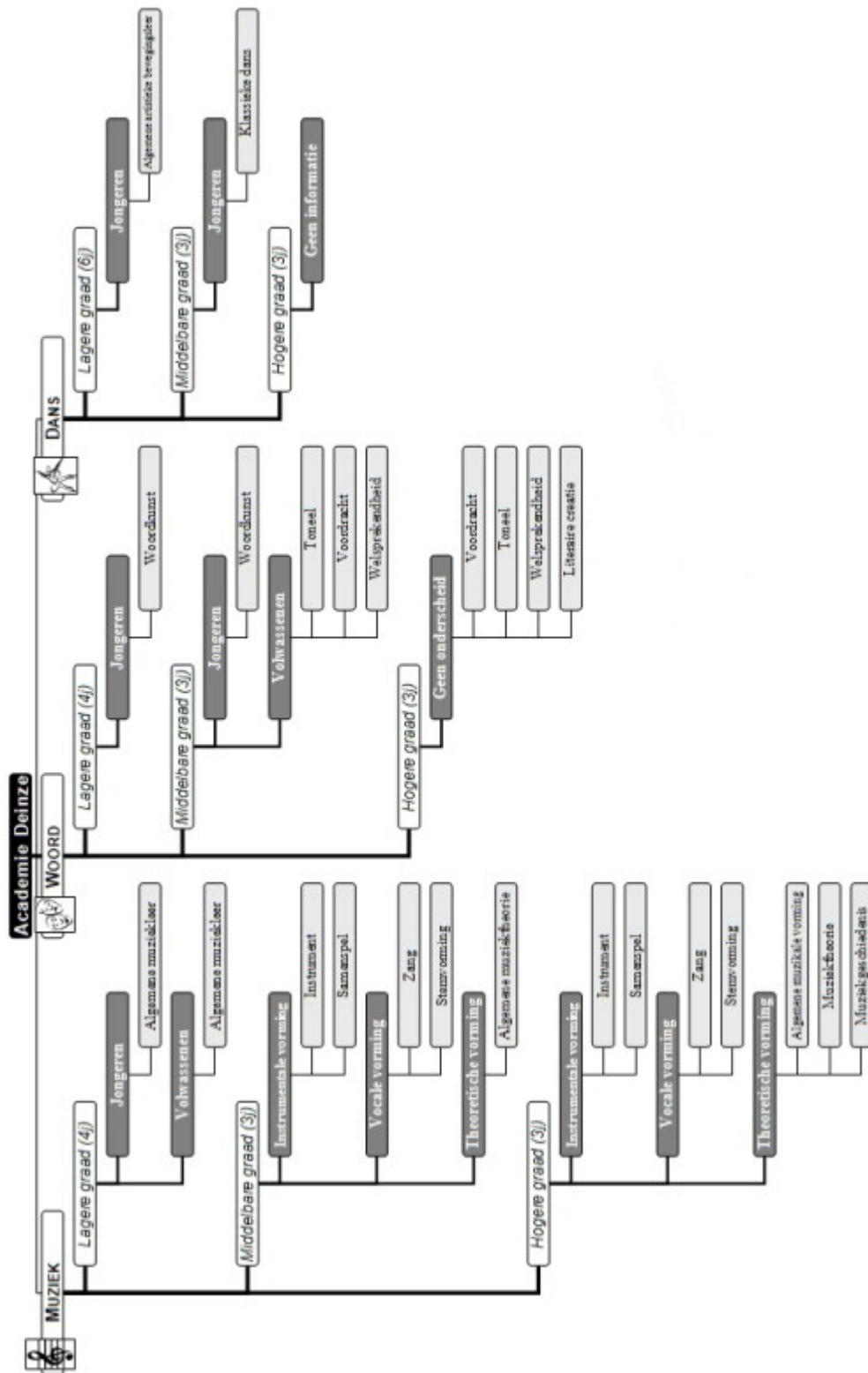
Na het opstellen van de beschrijvende tekst en het hiervan afleiden van een Entity/Relationship-diagram is het uitschrijven van het relationele schema de derde en finale stap in het databankontwerpproces. Hiervoor werd een algoritme in 7 stappen gebruikt (uit de cursus ‘data-

banktechnologie'), een soort handleiding die naargelang het relatie- of entiteitstype beschrijft welke relaties (tabellen) er voorzien dienen te worden en welke sleutels en attributen hieraan moeten toegekend worden. Zonder in detail te treden komt het er bij wijze van voorbeeld op neer dat binaire N:M relatietypes (zoals 'inschrijving') in stap 5 van deze procedure aanleiding zullen geven tot een aparte tabel, met als velden de attributen van dit relatietype (hier dus enkel 'vrijstelling') en als primaire sleutel de combinatie van de vreemde sleutels van de betrokken entiteitstypes (voor dit voorbeeld wordt de primaire sleutel dus: stamnummer + graadID + optieID + vakID + groepID) . In stap 6 wordt voor elk meerwaardig attribuut (zoals 'evaluatie_X') eveneens een nieuwe relatie gecreëerd. Hier dient men attributen toe te voegen die een waarde voor 'evaluatie_X' kunnen bevatten. Als bovendien dit meerwaardig attribuut nog eens samengesteld is (dit is het geval), dan worden er attributen toegevoegd voor elke enkelvoudige component. Het resultaat is de volgende tabelstructuur (de primaire sleutel van elke tabel werd gecursiveerd):

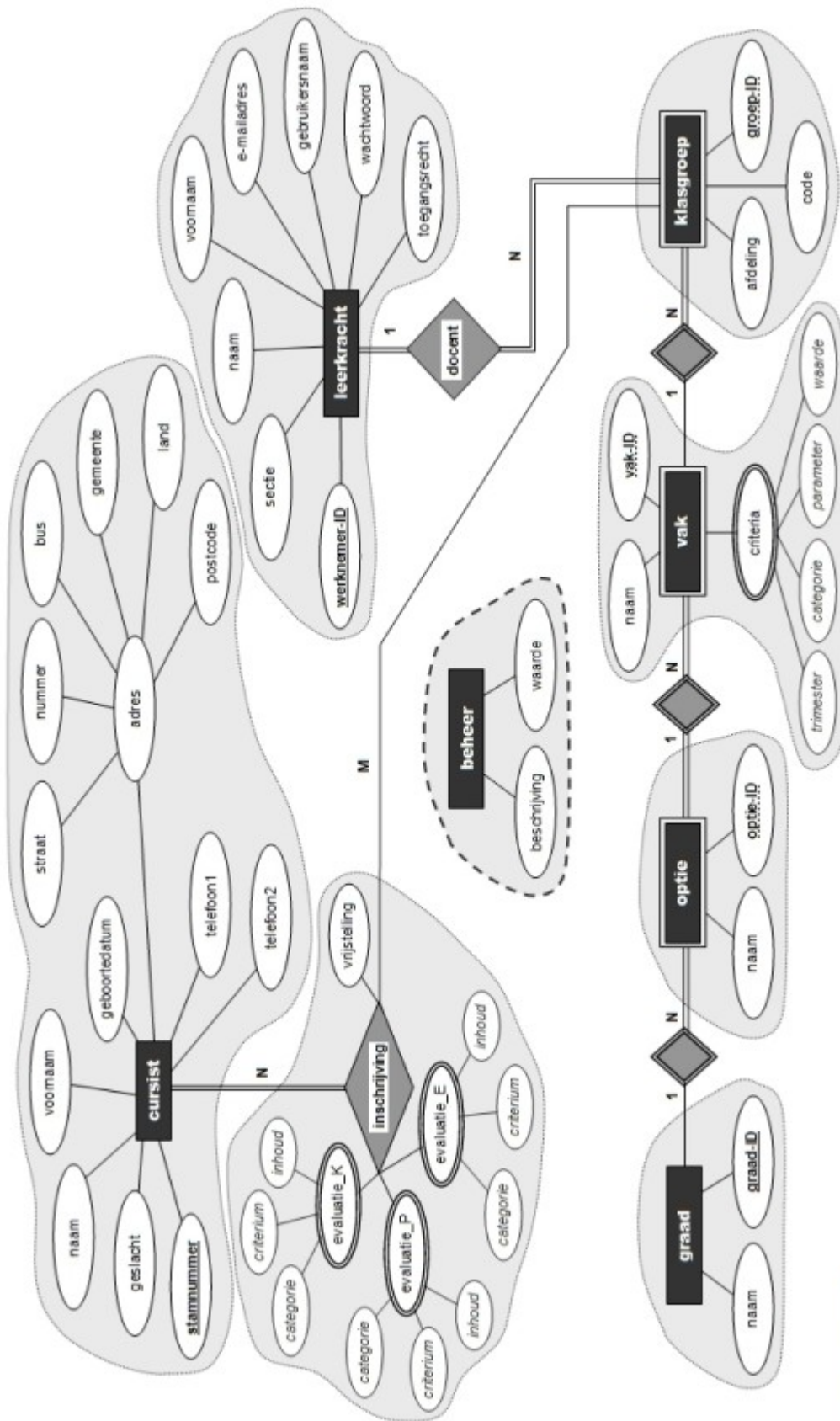
- **cursist**(*stamnummer,voornaam,naam,geslacht,geboortedatum,telefoon1,telefoon2,straat,nummer,bus,postcode,gemeente,land*)
- **leerkracht**(*werknemerID,sectie,voornaam,naam,emailadres,gebruikersnaam,wachtwoord,toegangsrecht*)
- **graad**(*graadID,naam*)
- **optie**(*graadID,optieID,naam*)
- **vak**(*graadID,optieID,vakID,naam*)
- **klasgroep**(*graadID,optieID,vakID,groepID,afdeling,code*)
- **inschrijving**(*stamnummer,graadID,optieID,vak ID,groepID,vrijstelling*)
- **docent**(*werknemerID,graadID,optieID,vakID,groepID*)
- **evaluatie_K**(*stamnummer,graadID,optieID,vakID,groepID,categorie,criterium,inhoud*)
- **evaluatie_P**(*stamnummer,graadID,optieID,vakID,groepID, categorie,criterium,inhoud*)
- **evaluatie_E**(*stamnummer,graadID,optieID,vakID,groepID, categorie,criterium,inhoud*)
- **criteria**(*graadID,optieID,vakID,trimester,categorie,parameter,waarde*)
- * **beheer**(*beschrijving,waarde*)

Men bekomt dus een totaal van 13 tabellen voor de nieuwe databank, waarmee men in principe alle behoeften van de website zou moeten kunnen ondersteunen en waarmee aan de verwachtingen van de gebruikers zou moeten voldaan zijn. Een opmerking dient hierbij gemaakt te worden: op het moment dat het grootste deel van het programmeerwerk van de

PHP-scripts achter de rug lag, was er de noodzaak om nog een wijziging in het ER-diagram aan te brengen (de unieke koppeling tussen docent en klasgroep werd geïntroduceerd door de kardinaliteitsratio van het relatietype ‘docent’ te wijzigen) om alle gevraagde functionaliteit te kunnen implementeren. Dit betekende concreet dat de tabel ‘docent’ in principe mocht weggelaten worden en vervangen door een bijkomende identifier in de tabel ‘klasgroep’. Deze wijziging werd echter niet meer doorgevoerd aangezien de tot daartoe gebruikte databankstructuur perfect voldeed. Maar in principe zou het hele systeem dus ook met nog een tabel minder kunnen verwezenlijkt worden. Daarnaast werd ook overwogen om deze tabelstructuur te normaliseren, om aldus redundantie te elimineren. Aangezien normalisatie echter hoofdzakelijk berust op decompositie (opsplitsing), werd besloten om dit niet te doen: dit zou de logica achter de tabelstructuur vermoedelijk teveel ‘verstopt’ hebben (minder transparant naar de gebruikers toe) en door de decompositie zouden meer ‘(inner) joins’ van tabellen nodig geweest zijn in de SQL-queries, hetgeen de bevragingen enkel had bemoeilijkt.



Figuur 4.1: Hiërarchische opdeling in secties (3 kolommen), graden (witte vakjes) en opties (lichtgrijze vakjes) van de opleidingen van de AMWD Deijnze.



Figuur 4.2: Entity/Relationship-diagram van de nieuwe databank voor de AMWD Deinze.

Deze tabelstructuur werd — zoals eerder vermeld — geïmplementeerd in MySQL door een testdatabank op te bouwen, die manueel opgevuld werd met realistische dummy-gegevens. Daarbij diende ook het MySQL-datatype van elk attribuut gespecificeerd te worden. Er werd speciale aandacht besteed aan het kiezen van het meest adequate datatype: enerzijds moest dit compatibel zijn met het type en de lengte van de inhoud van het attribuut en anderzijds werd erop gelet dat de benodigde geheugenruimte niet meer was dan strikt gezien noodzakelijk (bv. niet het datatype TEXT kiezen met een stringlengte van max. 65535 karakters voor de opslag van een string van 10 karakters). Een goede keuze kan de performantie van de databank namelijk verhogen. Bij datatypes voor getallen werd meestal gekozen voor de ‘UNSIGNED’-variant, wat de mogelijkheden beperkt tot de positieve getallen en nul. Soms konden ook standaardwaarden opgegeven worden (bv. ‘land’ zal bijna altijd ‘België’ zijn). Bij de meeste datatypes diende ook de maximale lengte (in aantal karakters of bytes; 1 karakter \approx 1 byte) opgegeven te worden.

We geven hierna een opsomming van alle voorkomende attributen met hun datatype en een (fictief) voorbeeld van een waarde (of enkele waarden) die ze kunnen aannemen. De ‘collatie’ is de tekenset die gebruikt wordt bij onder meer karakterstrings. Standaard werden subsets van de *latin*-collatie gebruikt. Soms was het echter nodig om enkel ASCII-tekens toe te laten.

Tabel 4.1: Attributen per tabel met hun datatype, collatie en voorbeeldwaarde(n).

Attribuut	Datatype	Mogelijke inhoud	Collatie
cursist			
stamnummer	smallint(5) unsigned	5096	-
naam	varchar(35)	Peeters	stand.
voornaam	varchar(35)	Jan	stand.
geslacht	set(‘M’,‘V’)	M	stand.
geboortedatum	date	1994-02-18	-
telefoon1	varchar(20)	09/392.18.26	stand.
telefoon2	varchar(20)	0496/23.18.88 (default: NULL)	stand.
straat	varchar(70)	Molenstraat	stand.
nummer	smallint(4)	7	-
bus	varchar(5) unsigned	2A (default: NULL)	stand.
postcode	smallint(4) unsigned	9000	-
gemeente	varchar(35)	Gent	stand.
land	varchar(20)	België (default)	stand.
leerkracht			
werknemerID	tinyint(3) unsigned	9	-

sectie	set('Muziek','Woord','Dans')	Muziek	stand.
voornaam	varchar(35)	Stefaan	stand.
naam	varchar(35)	Cornelus	stand.
emailadres	varchar(70)	stefaan.cornelus@adsl.be	ASCII
gebruikersnaam	varchar(70)	scornelus	ASCII
wachtwoord*	varchar(35)	pf7b36x	ASCII
toegangsrecht	set('A','U','V')	A	stand.
graad, optie, vak			
graadID	tinyint(3) unsigned	5	-
optieID	tinyint(3) unsigned	3	-
vakID	tinyint(3) unsigned	5	-
naam	varchar(x) x = 35 of 70	muziek middelbare graad 1	stand.
		instrument	stand.
		klarinet	stand.
klasgroep**			
groepID	tinyint(3) unsigned	2	-
afdeling	varchar(70)	Kruishoutem	stand.
code	varchar(10)	ASC of BSC of NOSC of MIXSC	-
inschrijving			
vrijstelling***	set('0','50','100')	0	stand.
criteria			
trimester	set('K','P','E','J','nvt')	P	stand.
categorie	varchar(70)	nvt	
		creativiteit	stand.
		G, O of T	
parameter	varchar(70)	fantasie	
		totaal	stand.
		NULL	
		dagel.werk	
		tekst	
waarde	varchar(70)	niveau	
		code	stand.
		gradatie	
		20	
		NULL	
evaluatie_K, evaluatie_P en evaluatie_E			
categorie	varchar(70)	nvt	
		creativiteit	stand.
		examen	

criterium	varchar(70)	fantasie geen totaal dagel. werk	stand.
inhoud	text	goed gewerkt! 18 a nvt	stand.
beheer			
beschrijving	varchar(70)	evaluatiemoment mededeling1 deadline_rapport	stand.
waarde	text	1 Vrijdag vergadering 24/12/2006	stand.

* in de tabel ziet men enkel de MD5-geëncrypteerde versie van het wachtwoord 'pf7b36x'

** de tabel 'docent' maakt gebruik van attributen die ook voorkomen in de tabellen 'klasgroep' en 'leerkracht'. Dergelijke gemeenschappelijke attributen worden slechts één maal in de tabel opgenomen

*** bij vrijstelling krijgt de groepID waarde 'null' ('leeg') en wordt 'vrijstelling' 50 of 100

Concrete gebruik van de nieuwe databank

Aan de hand van Tabel 4.1 en Figuur 4.2 zal nu de wijze waarop van de nieuwe databank gebruik wordt gemaakt verduidelijkt worden. Dit zal meer klaarheid scheppen omtrent de manier waarop alle essentiële gegevens in de databank worden opgenomen en hoe deze zich tot elkaar verhouden. In de volgende paragrafen zullen dan de scripts besproken worden, die van dit model gebruik maken om de databank te manipuleren (data toevoegen, aanpassen of verwijderen) of te bevragen.

De tabel ‘cursist’ doet dienst als opslagplaats voor die administratieve gegevens van de studenten die nodig zijn om de evaluatie te kunnen uitvoeren en om het rapport weer te geven of af te drukken. Voor de drie bestaande secties (muziek, woord en dans) zal een apart Excel-bestand aangemaakt worden dat alle noodzakelijke gegevens (naam, voornaam, volledig adres, geslacht, geboortedatum en telefoonnummers) bevat per student. Daarnaast dient deze file ook de inschrijvingsgegevens per student te bevatten (in welke graad, optie, vak en klasgroep heeft de student zich ingeschreven? is hij/zij voor het ingeschreven vak vrijgesteld? bij welke leerkracht en in welke afdeling wordt het vak gevolgd? ...). De uitgangsbasis is dus nog steeds dezelfde: het secretariaat van de academie voert alle inschrijvingsgegevens van de studenten bij het begin van een nieuw schooljaar in via het beschikbare computerprogramma (=data-administratie); op basis hiervan selecteert de beheerder de noodzakelijke velden en maakt hij CSV-bestanden aan, die gebruikt worden om de databank op te vullen (=databank-administratie). Aan de vraag om niet aan het huidige invoersysteem te raken (cfr. paragraaf 2.2) is bijgevolg voldaan. Wel wordt gevraagd om voortaan niet meer gebruik te maken van de oude letter- en cijfercodes (tenzij voor eigen intern gebruik), maar wel van een nieuw type code die nog enkel informatie bevat omtrent de afdeling, de klasgroep en de leerkracht. Door deze drie CSV-bestanden dan via de nieuwe website te uploaden (privilege van de beheerder!) zullen de tabellen ‘cursist’ en ‘inschrijving’ automatisch gegenereerd en opgevuld worden voor het nieuwe schooljaar. Onderstaand voorbeeld (gebaseerd op Tabel 4.1) geeft het verschil aan tussen de oude en nieuwe manier van werken:

Veronderstel dat Jan Peeters, een jongeman van 12 jaar die reeds 4 jaar academie achter de rug heeft (‘notenleer’ en klarinet), zich tijdens één van de voorbije inschrijvingsdagen heeft ingeschreven voor het vak ‘klarinet’ voor het volgende jaar (middelbare graad 1, optie ‘instrument’, leerkracht: Stefaan Cornelus, afdeling Kruishoutem). Zijn keuze alsook zijn andere gegevens werden door het secretariaat aan het Excel-bestand voor de sectie muziek toegevoegd.

In het oorspronkelijke systeem zagen de CSV-velden er dan als volgt uit:

naam	voornaam	straatnaam	huisnr	busnr	postcode	gemeente
Peeters	Jan	Molenstraat	7	2A	9000	Gent
land	geboorteda	telefoon1	telefoon2	geslacht	optiekort	klasgroep
België	1994-02-08	09/392.18.26	0496/23.18.88	M	INS	KLSC
stamnummer	graad	code	leerjaar	vaknaam	leraar	naamoptie
5096	MMG	SCKLKROI	MM1	klarinet	Cornelus Stefaan	instrument
sectie						
M						

In het nieuwe systeem zijn er minder velden vereist. Dit komt omdat een stuk van de redundantie die in de verschillende oude codes aanwezig was, geëlimineerd kon worden. Men bekomt aldus:

naam	voornaam	straatnaam	huisnr	busnr	postcode	gemeente
Peeters	Jan	Molenstraat	7	2A	9000	Gent
land	geboorteda	telefoon1	telefoon2	geslacht	stamnummer	graad
België	1994-02-08	09/392.18.26	0496/23.18.88	M	5096	MM1
optie	vak	klasgroep	leraar			
instrument	klarinet	KASC	Cornelus Stefaan			

De velden ‘optiekort’, ‘klasgroep’, ‘graad’ en ‘sectie’ hoeven dus niet meer ingevuld (of vanuit de Excel-file geselecteerd) te worden voor de ondersteuning van het centraal evaluatie- en rapporteringssysteem. Al naargelang men de oude velden voor intern gebruik wenst te behouden of niet, betekent dit dus evenveel of minder werk voor de mensen van het secretariaat. De nieuwe velden zullen vertaald worden naar een graadID, optieID, vakID en groepID, die dan verder kunnen gebruikt worden in bevragingen van de databank. De enige échte verandering situeert zich in het oude veld ‘code’. Om compatibel te zijn met het nieuwe systeem dient hier een andere codering gehanteerd te worden om zo aanleiding te geven tot het nieuwe veld ‘klasgroep’. Daar waar ‘code’ informatie bevatte over de leerkracht (SC), *het vak* (KL) en de afdeling (KRUI), zal ‘klasgroep’ informatie bevatten over de afdeling (K), *de groep* (A) en de leerkracht (SC). Deze nieuwe code zegt dus niets meer over het gevolgde vak (daar dit al uit een ander veld kan afgeleid worden), maar eerder over de klasgroep waarin de leerling zich bevindt.

De nieuwe code ‘klasgroep’ bestaat dus uit drie delen. De mogelijke combinaties staan weergegeven in Tabel 4.2. Het eerste deel, de afdeling, werd consequenter gecodeerd dan voorheen (slechts 1 letter per afdeling). Daarna volgen één of meerdere letters die de groep aangeven. Wanneer een vak van een zekere leerkracht in een bepaalde graad gevolgd wordt door een groot aantal studenten, dan kan de leerkracht beslissen om deze studenten op te delen in een aantal groepen (logisch benoemd door A, B, C, enz.). Indien er enkel een opdeling is in jongeren en volwassenen, dan kunnen de letters J en V gebruikt worden. Is er echter slechts één groep met leerlingen uit dezelfde graad, dan wordt dit aangegeven door NO (‘niet opgesplitst’). Het komt ook voor dat leerkrachten — bij de evaluatie — geen bijzonder onderscheid wensen te

maken tussen leerlingen uit verschillende graden die hetzelfde vak volgen. In dat geval wordt de groep aangeduid door MIX. Door met deze letter(combinatie)s te werken, kunnen bij de weergave van de klasgroepen van een leerkracht de groepen hiërarchisch opgedeeld worden, respectievelijk volgens vak, afdeling, graad en groep. Dit zal het gebruiksgemak van de website verhogen. Ten slotte worden ook de initialen van de leerkracht toegevoegd. Dit is nodig voor deze situaties waarbij twee verschillende leerkrachten hetzelfde vak zouden onderwijzen in dezelfde graad, dezelfde afdeling en in hetzelfde type groep (bv. beide een MIX-klas of beiden klas B). Het is dan ook niet nodig om bijvoorbeeld aan de ene leerkracht klassen A en B toe te wijzen en aan de andere leerkracht klassen C en D: elk kan voortaan z'n eigen 'nummering' gebruiken, onafhankelijk van andere leerkrachten.

Tabel 4.2: Samenstelling van de code 'klasgroep'.

afdeling	groep	leerkracht
D (Deinze)	A	SC
K (Kruishoutem)	B	MB
L (Sint-Martens-Latem)	C	GDS
N (Nazareth)	...	SVH
Z (Zulte)	J	MM
...	V	PV
...	NO	...
...	MIX	...

Studenten worden gekarakteriseerd door hun unieke 4-cijferige stamnummer. Leerkrachten worden dan weer aangeduid door een 'werknemerID'. De vrij algemeen gekozen benaming van deze primaire sleutel wijst er al op dat ook andere leden van de academie eventueel in deze tabel kunnen opgenomen worden, zodat ze eveneens toegang krijgen tot (delen van) de website. Het attribuut 'toegangsrecht' (met als mogelijke waarden 'administrator', 'user' en 'visitor') zorgt voor deze vorm van 'discretionary control' (die zich situeert ter hoogte van de welkompagina van de gebruikerswebsite). Voor het gebruik van de tabel 'beheer' verwijzen we naar de bespreking van de gebruikerswebsite.

De tabellen 'criteria', 'evaluatie_K', 'evaluatie_P' en 'evaluatie_E' bevatten een groot deel van de 'intelligentie' van de databank. Hier diende immers meer flexibiliteit en functionaliteit ingebouwd te worden. In de tabel 'criteria' worden de door de leerkracht opgegeven beoordelingsparameters opgeslagen per vak. Maar ook de modaliteiten betreffende de examens worden hier vastgelegd. Dit kan het best geïllustreerd worden aan de hand van een concreet voorbeeld:

Leerkracht X geeft het vak 'algemene muzikale vorming' (vakID=3) in de lagere graad 1 (graad-

ID=1). Dit vak valt onder de optie ‘algemene muziekleer’ (optieID=1). Samen met zijn/haar collega’s Y en Z, die hetzelfde vak geven in dezelfde graad, heeft leerkracht X in het begin van het schooljaar afgesproken om de leerlingen te evalueren op 11 criteria. Drie van deze criteria dienen beoordeeld te worden a.d.h.v. een cijfer, de andere criteria worden beoordeeld met een tekstje. Vanaf volgend jaar zullen de leerkrachten X, Y en Z een klein partieel examen voorzien vóór de Kerstvakantie, maar dit jaar zullen ze — zoals de voorbije jaren gangbaar was — geen examen geven rond Kerst. Rond Pasen wordt wel een partieel examen voorzien. Twee cijfers zullen gegeven worden: dagelijks werk op 30 en de punten van het theorie-examen, op 70. Voor het einde van het jaar willen ze het eenvoudiger houden: één cijfer op 100 moet dan volstaan voor het eindexamen.

Deze voorwaarden zullen in de tabel ‘criteria’ als volgt worden gedefinieerd (niet alle records worden hier afgebeeld):

Tabel 4.3: Voorbeeld van de inhoud van de tabel ‘criteria’.

graadID	optieID	vakID	trimester	categorie	parameter	waarde
1	1	3	nvt	nvt	alg. kennis	20
1	1	3	nvt	nvt	gehoor	20
1	1	3	nvt	nvt	houding	tekst
1	1	3	nvt	nvt	juiste toon zingen	tekst
1	1	3
1	1	3	nvt	nvt	lezen van noten	tekst
1	1	3	K	G	<i>null</i>	<i>null</i>
1	1	3	P	O	dagelijks werk	30
1	1	3	P	O	theorie	70
1	1	3	E	T	totaal	100

Hierbij staan de letters G, T en O voor resp. ‘geen examen’, ‘totaal’ en ‘opgesplitst’. De *null*-waarden betekenen concreet dat deze velden (cellen) open gelaten worden in de Excel-file die gebruikt wordt om de tabel ‘criteria’ op te vullen (via de upload-procedure). Maar nog meer flexibiliteit is mogelijk: leerkrachten die hun criteria wensen op te delen in een aantal categorieën, dienen dit gewoon in de kolom ‘categorie’ op te geven, zoals hieronder aangegeven voor de categorieën ‘theorieles’ en ‘praktijk’. Deze opdeling zal te zien zijn op de evaluatiepagina’s en op het rapport.

In de tabellen ‘evaluatie_X’ (voor de drie evaluatiemomenten) worden dan de resultaten van deze beoordelingen gestockeerd per leerling (opnieuw van zowel de criteria als van de examens). In Tabel 4.5 wordt het resultaat van het invullen van de criteria uit Tabel 4.3 en 4.4 gegeven (voorbeeld mét categorieën). Merk op dat de laatste vier records van deze tabel normaalgezien niet tesamen voorkomen, maar elk in de juiste tabel (corresponderend met

Tabel 4.4: Zelfde voorbeeld als Tabel 4.3, mét categorieën.

graadID	optieID	vakID	trimester	categorie	parameter	waarde
1	1	3	nvt	theorieles	alg. kennis	20
1	1	3	nvt	praktijk	gehoor	20
1	1	3	nvt	theorieles	houding	tekst
1	1	3	nvt	praktijk	juiste toon zingen	tekst
1	1	3
1	1	3	nvt	praktijk	lezen van noten	tekst

Kerst, Pasen of het einde van het jaar). Het totaal dat weergegeven wordt (vanwege de beoordeling met één cijfer op het einde van het jaar, uit het voorbeeld) stelt een *gegeven* cijfer voor. Wanneer men — zoals rond Pasen — een aantal cijfers geeft, dan wordt hiervan ook het totaal berekend. Dit *berekende* totaal komt dan op dezelfde manier (zoals hier: $26+54=80$) als bij een gegeven cijfer in de tabel terecht.

Tabel 4.5: Voorbeeld van de inhoud van de tabel ‘evaluatie_X’.

stamn	gra.ID	opt.ID	vakID	gro.ID	categorie	criterium	inhoud
3421	1	1	3	8	theorieles	alg. kennis	17
3421	1	1	3	8	praktijk	gehoor	15.5
3421	1	1	3	8	theorieles	houding	Werkt goed mee
3421	1	1	3	8	praktijk	juiste toon zingen	Blijven oefenen!
3421	1	1	3	8
3421	1	1	3	8	praktijk	lezen van noten	Gaat vlot
3421	1	1	3	8	examen	geen	geen partieel examen
3421	1	1	3	8	examen	dagelijks werk	26
3421	1	1	3	8	examen	theorie	54
3421	1	1	3	8	examen	totaal	80

Een kleine opmerking hierbij: wanneer men gebruik maakt van een O-type beoordeling (zoals in Tabel 4.5 met ‘dagelijks werk’ en ‘theorie’ als categorieën), dan dient men de namen van deze categorieën zo te kiezen dat er geen operatortekens in voorkomen. Deze bewerkingstekens (zoals +, -, *, /, %) zouden de automatische weergave van het totaal (zie verder) immers beïnvloeden, wat uiteraard niet gewenst is. Men moet er dus vooral op letten bij het kiezen van de categorienamen dat men bv. geen koppelttekens gebruikt. Vandaar dus ‘theorie’ op 70 punten en *niet* ‘theorie-examen’ op 70 punten (zie Tabel 4.3)!

Op deze manier kan een grote hoeveelheid relatief complexe beoordelingsdata op een logische manier ondergebracht worden in drie tabellen (scheiding van de evaluatiemomenten), afgezonderd van de criteria die in een aparte tabel worden bijgehouden. Deze manier van werken

vraagt iets ingewikkeldere PHP-scripts, maar zorgt voor een compacte en logisch opgebouwde databank.

4.2 De databankbeheerpagina's

4.2.1 Overzicht van de beheerpagina's

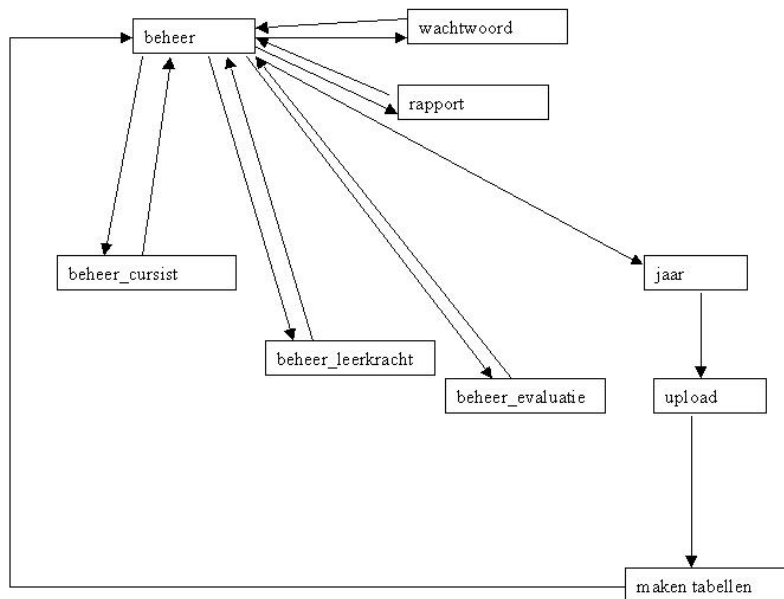
Als de gebruiker administratorrechten heeft kan die via de vanaf de centrale welkompagina de beheerpagina's bereiken. Daar kan de administrator alle handelingen stellen die nodig zijn om de databank te beheren en aan te maken. Eenmaal de administrator naar de beheerpagina's gaat komt hij op een centrale administratiepagina terecht. Een screenshot is te zien in figuur 4.3. Vanaf deze pagina heeft de administrator de keuze tussen verschillende handelingen. Een schematisch overzicht is te vinden in figuur 4.4. Daarin is te zien dat de beheerder vanaf de centrale pagina het wachtwoord van iedere gebruiker kan wijzigen, een rapport aanmaken en een nieuwe database aanmaken. Ook kan hij een bestaande database kiezen. Daar kan de administrator kiezen uit de tabel cursist, leerkracht en evaluatie.



Figuur 4.3: Screenshot van de centrale beheerpagina

De pagina voor het wijzigen van het wachtwoord genereert een random wachtwoord voor een bepaalde gebruiker die zijn wachtwoord vergeten is. Ook kan de administrator daar zelf zijn wachtwoord wijzigen. Weliswaar zal hij geen keuze hebben hoe zijn wachtwoord eruit ziet, het zal een random wachtwoord zijn. Wil hij zelf kunnen kiezen, dan dient hij dezelfde weg te volgen als een gewone gebruiker. Bovenaan de pagina staat er een knop waarmee hij zijn wachtwoord kan wijzigen. De pagina om een rapport aan te maken genereert een LaTeX bestand. De administrator heeft de keuze uit 1 rapport voor een bepaalde student of uit alle rapporten voor een heel academiejaar. Als de beheerder kiest uit een bestaande database krijgt hij daarna de keuze uit de pagina beheer_cursist, beheer_leerkracht en beheer_evaluatie.

Als de pagina beheer_cursist gekozen wordt kan de administrator daar gegevens van studenten veranderen. De velden die kunnen veranderd worden zijn adres en telefoon. Alle niet statische gegevens van een persoon dus. Hierbij gaan we er van uit dat een persoon zijn naam niet wijzigt. Indien het om een of andere reden toch nodig mocht zijn om een van de



Figuur 4.4: Schematisch overzicht van de beheerpagina's

statische gegevens te wijzigen, wegens bijvoorbeeld een fout, dan kan de administrator dat altijd rechtstreeks doen in de tabel cursist.

Als de pagina beheer leerkracht gekozen wordt kan de beheerder daar volgende gegevens wijzigen: het emailadres van de docent en de gebruikersnaam. Deze wijzigingen worden dan door het script in de tabel leerkracht toegepast. Het wijzigen van het wachtwoord werd niet voorzien, aangezien er al een pagina is die specifiek daartoe bestaat.

Kiest men de pagina beheer_ evaluatie dan kunnen daar alle velden gewijzigd worden. Dit is vooral nuttig als een evaluatieperiode reeds is afgesloten. Docenten kunnen eenmaal een evaluatieperiode voorbij is, geen wijzigingen meer aanbrengen voor de voorbije perioden. Als een docent een foute evaluatie ingeeft en dit wordt pas opgemerkt als de evaluatie voorbij is, kan de administrator toch nog de gegevens aanpassen.

Keren we terug naar de centrale beheerpagina, dan blijkt dat we de optie een nieuwe database aanmaken nog niet bekeken hebben. Als daarvoor gekozen wordt kan de beheerder dus een nieuwe databank aanmaken. Eerst krijgt de administrator de mogelijkheid om een academie-

jaar voor de databank op te geven. Deze zal dan de naam eval0x0y krijgen, waarbij 0x0y voor het desbetreffende academiejaar staat. Eenmaal dat gekozen is er de mogelijkheid om bestanden up te loaden. Deze bestanden zullen gebruikt worden om de tabellen op te vullen met gegevens. Het uploaden gebeurt via het HTTP protocol. Als de bestanden upgeload zijn wordt de databank aangemaakt en de tabellen ingevuld. Deze zaken gebeuren automatisch door het script. De beheerder ziet daar niets van. Als het aanmaken van de tabellen gelukt is krijgt de beheerder een bericht dat alles verlopen is zoals het moet en de mogelijkheid naar de centrale beheerpagina terug te keren. Indien er ergens een fout optreedt krijgt de administrator een foutmelding te zien.

In wat volgt worden de verschillende pagina's in detail besproken. Wat de scripts doen en hoe ze het doen.

4.2.2 De centrale beheerpagina

In deel 4.2.1 is de centrale beheerpagina al een vrij groot detail besproken. Ook is een screenshot te vinden in figuur 4.3. Wat nog niet echt in detail is besproken is hoe de beheerder een bestaande database kan kiezen. Ook is er niet vermeld dat de daarop volgende keuzemogelijkheden van beheer_ -cursist, leerkracht en evaluatie na de keuze van de database op de centrale beheerpagina verschijnt. Deze zijn voor de gekozen database. Indien de gebruiker een andere database wil bekijken, dient die gewoon een nieuwe te kiezen en de links naar de beheerpagina's van de tabellen cursist, leerkracht en evaluatie worden daar aan aangepast. Het keuzemenuutje voor de beschikbare databanken wordt als volgt opgebouwd. Eerst wordt via de PHP functie `mysql_list_dbs` alle beschikbare databanken in een array geplaatst. Daarna wordt via de functie `mysql_num_rows` het aantal databanken nagegaan. Een while lus loopt dan de array af en plaatst de aanwezige databanken in een list. De code ziet er als volgt uit.

```

$db_list = mysql_list_dbs($db);
$i = 0;
$cnt = mysql_num_rows($db_list);
while ($i < $cnt)
{

    $naamdb = mysql_db_name($db_list, $i);
    $string = substr($naamdb, 0, 4);
    if ($string=="eval")
    {

        echo "<option>".$naamdb."<option>";
    }
}

```

```

    }
    $i++;

}

```

Aangezien veiligheid bij het kiezen van de database geen rol speelt wordt via de GET gewerkt. Veiligheid speelt geen rol aangezien enkel een jaartal wordt gekozen. Ook is de beheerpagina niet beschikbaar als de gebruiker geen administratorrechten heeft. Eenmaal een database gekozen wordt de verborgen variabele "gekozen" geïnitieerd. Dit is van belang om te weten of er reeds een jaartal gekozen is. Indien dit niet het geval is, dan mogen de beheer-, cursist-, leerkracht en evaluatie linken niet zichtbaar zijn. Dit wordt nagegaan via de PHP functie `isset()`. Als er gekozen is worden de links naar de andere beheerpagina's weergegeven. Het jaartal wordt als een GET variabele meegegeven in de links.

4.2.3 Wachtwoord

Het wijzigen van het wachtwoord gebeurt door een simpele druk op de knop. Het eerste dat de beheerder te zien krijgt is een keuzelijst met alle gebruikers. Eenmaal gekozen voor een bepaalde gebruiker krijgt de beheerder de mogelijkheid te zien om het wachtwoord van de gekozen gebruiker te vernieuwen. Als de administrator het wachtwoord wijzigen bevestigt, dan wordt er automatisch een wachtwoord gegenereerd en door gemaild naar de desbetreffende gebruiker. Eenmaal dit gedaan, kan de administrator een andere gebruiker kiezen op dezelfde wijze of beslissen om terug te keren naar de centrale administratiepagina.

De code voor het keuzemenuutje van de gebruikers wordt op dezelfde wijze opgebouwd als het menuutje voor het kiezen van een database. Eerst wordt er in de tabel leerkracht alle gebruikersnamen in een array gestopt. Deze wordt dan op dezelfde wijze afgelopen in een `while` lus.

De random generator voor het wachtwoord ziet er als volgt uit. Er wordt een array aangemaakt bestaande uit het alfabet, alle cijfers en een aantal leestekens. Deze array wordt door elkaar gegooid, zodat men een array krijgt die random bestaat uit letters cijfers en leestekens. Van die array worden de eerste zeven elementen genomen als wachtwoord. Eenmaal dit gedaan is, wordt het wachtwoord naar de gebruiker gemaild en in de tabel opgeslagen. De code ziet er als volgt uit.

```

$keuze = array_merge(range('a','z'), range('0','9'), array("?", "+", "!", "<", "&", "%"));
shuffle($keuze);
$randomwacht = $keuze[0].$keuze[1].$keuze[2].$keuze[3].$keuze[4].$keuze[5].$keuze[6];

```

4.2.4 Beheer_cursist

Als de beheerder kiest voor de beheer_cursist pagina krijgt hij twee keuzemenuutjes te zien. Het ene is een lijst met alle namen van de cursisten, het andere is een lijst met alle stamnummers van de cursisten. Zo heeft de administrator de keuze om te kiezen volgens stamnummer of naam. Eenmaal gekozen krijgt de beheerder het stamnummer, de naam, het adres en de telefoonnummers van de gekozen te zien. Enkel het adres en de telefoonnummers zijn wijzigbaar. Na het wijzigen kan de beheerder een nieuwe cursist kiezen of terugkeren naar de centrale beheerpagina. De keuzemenuutjes wordt op gelijkaardige wijze opgebouwd als het menuutje van de wachtwoord pagina. Via een MySQL SELECT query worden de nodige velden geselecteerd om zo de gegevens van de cursist weer te geven.

4.2.5 Beheer_leerkracht

De beheerleerkrachtpagina heeft dezelfde opbouw als de beheer_cursist pagina. Het verschil is dat er slechts één keuzemenuutje is, met de naam van alle docenten, en dat andere velden wijzigbaar zijn. Als er een docent gekozen is verschijnen volgende velden: naam van de docent, zijn emailadres en gebruikersnaam. Enkel het emailadres en de gebruikersnaam zijn wijzigbaar. Eenmaal een wijziging aangebracht kan de beheerder kiezen een andere docent zijn gegevens te wijzigen of terug te keren naar de algemene beheerpagina. De code is vergelijkbaar met deze voor het beheer van de cursist. Er worden in de query enkel andere attributen geselecteerd, namelijk de naam, voornaam, gebruikersnaam en het emailadres.

4.2.6 Beheer_evaluatie

Als voor beheer_evaluatie gekozen wordt krijgt men eerst drie keuzemenuutjes te zien. In het ene wordt de naam van de cursist gekozen, in het ander kan eventueel het stamnummer gekozen worden en via het laatste kiest men het evaluatiemoment, zijnde Kerst, Pasen of Eind. Eenmaal gekozen worden voor de betreffende student en het gekozen evaluatiemoment alle resultaten weergegeven in een tabel. In die tabel zijn alle velden wijzigbaar. Waar er bij de evaluatie voor de docenten keuzemogelijkheden waren zijn die ook beschikbaar via een menuutje. Deze keuzemogelijkheden worden, zoals bij de gebruikerswebsite, uit de tabel criteria gehaald. Eenmaal de wijzigingen doorgevoerd kan de beheerder wederom kiezen om ofwel van een ander evaluatiemoment en/of cursist de resultaten te wijzigen of terug te keren naar de centrale beheerpagina.

4.2.7 Rapport

Bij de rapportpagina is er eerst de keuze één rapport aanmaken of alle rapporten aanmaken. De keuze wordt doorgegeven via GET. Als er gekozen wordt voor één rapport dan heeft men daarna de keuze uit een lijst cursisten. Eenmaal gekozen wordt er daarna het rapport in

LaTeX formaat aangeboden om op te slaan op de gebruiker zijn computer. Indien er gekozen wordt om alle rapporten af te drukken dan worden onmiddellijk alle rapporten aangeboden in één LaTeX bestand. Dit dient de gebruiker dan zelf nog te compileren tot het voor hem te kiezen formaat, bijvoorbeeld PDF. Het opstellen van de rapporten verloopt op een zodanige manier dat eerst de jaren en graden in opklimmende volgorde doorloopt. Voor iedere cursist worden bij de eerste keer dat hij tegengekomen wordt alle vakken in één rapport gestoken. Dus als hij vakken in verschillende graden en jaren volgt, worden die bij de eerste opbouw weergegeven. Op die manier zijn de rapporten gerangschikt per klas jaar en graad. Ook wordt zo vermeden dat sommige rapporten meermaals worden afgedrukt. Na het opbouwen van de bestanden heeft de beheerder de keuze om een andere student te kiezen of terug te keren naar de algemene beheerpagina.

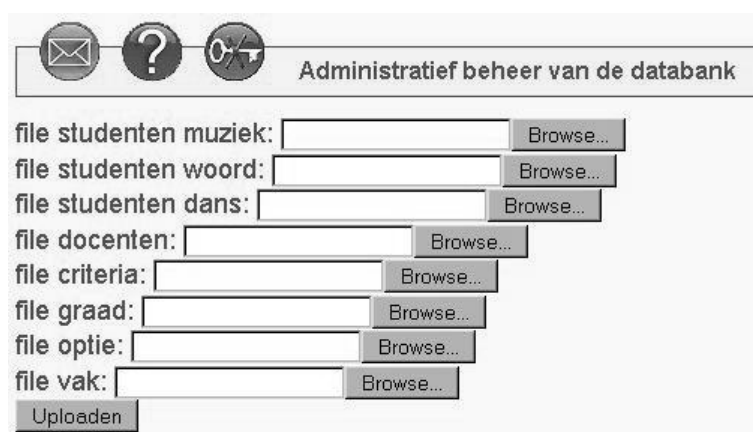
4.2.8 Het uploaden van de gegevens

Als de beheerder kiest om een nieuwe database aan te maken krijgt hij eerst een pagina te zien waarop hij het academiejaar van de databank kan ingeven. Een screenshot is te zien in figuur 4.5. Eenmaal het jaar gekozen, krijgt men de pagina te zien voor het uploaden van



Figuur 4.5: Screenshot van de pagina waar het academiejaar ingegeven wordt.

de gegevens. Een screenshot is te zien in figuur 4.6. Op deze pagina kan men verschillende



Figuur 4.6: Pagina voor het uploaden van de bestanden.

bestanden meegeven. Dit zijn de CSV bestanden. Ze hebben de gegevens van de studenten

muziek, woord en dans. Van de docenten, de criteria, alle graden, opties en vakken.

Het CSV bestand van de studenten muziek, woord en dans heeft volgend stramien:

```
stamnummer;voornaam,familienaam;geslacht;geboortedatum;telefoon1;telefoon2;
straat;huisnummer;postbus;postcode;woonplaats;land;klasgroep;graad;vak;docent;filiaal
```

Hierbij heeft de geboortedatum volgende syntax: jjj-mm-dd. De graad wordt weergegeven door de eerste letter van de sectie; zijnde muziek, woord of dans; gevolgd door de eerste letter van de graad; zijde lager,middelbare of hoger; en een cijfer dat het leerjaar representeert; zijnde 1,2,3,... Dit alles in hoofdletters. Bijvoorbeeld muziek lagere graad 2 wordt ML2. De naam van de docent wordt als volgt weergegeven. Eerst wordt de familienaam geplaatst, daarna de voornaam. Alles wordt is in hoofdletters en de naam van de docent bestaat maximaal uit 17 karakters, spaties inbegrepen.

Het CSV bestand van de docenten heeft volgend stramien:

```
sectie;voornaam;familienaam;emailadres;gebruikersnaam;rechten
```

Hierbij kan de sectie Muziek, Woord of Dans zijn en zijn de rechten ofwel U ofwel A. U staat voor gewone gebruiker en A voor administrator.

Het CSV bestand van de criteria heeft volgend stramien:

```
graad;optie;vak;periode;categorie;parameter;waarde
```

Indien een bepaald veld geen inhoud heeft wordt er nvt (niet van toepassing) geplaatst. Hoe de verschillende attributen eruit moeten zijn werd reeds besproken in 4.1.

Het CSV bestand van graad heeft volgend stramien:

```
;graad
```

Hierbij is de graad voluit geschreven. Bijvoorbeeld muziek lagere graad 1.

Het CSV bestand voor optie ziet eruit als:

```
;optie
```

Het CSV bestand voor vak ziet eruit als:

```
graad;optie;vak
```

Bij het uploaden moet men erop letten dat in de naam van de input tag file geen punten staan. Want dit geeft problemen bij de FILES variabele.

4.2.9 Het aanmaken van de tabellen

Voor we met het aanmaken van de tabellen kunnen beginnen moeten we eerst de upgeloade bestanden verplaatsen in onze map. Daartoe moeten we weten waar de server ze geplaatst

heeft. Dit kan eenvoudig gevonden worden via de variable `$_FILES`. Eenmaal het pad van de bestanden gekend is worden ze gekopieerd in onze map. Hier zit wel een addertje onder het gras. Namelijk de permissies. PHP wordt gerund door een bepaalde daemon die geen schrijfrechten heeft in onze map. Moesten we dus gewoon kopiëren, dan krijgen we een foutmelding. Dit kan opgelost worden door onze map open te stellen voor iedereen. Dit is natuurlijk een serieus veiligheidsrisico. Daartoe wordt er manueel een map aangemaakt waarin iedereen uitvoerrechten heeft. In die map wordt er een andere map aangemaakt die voor iedereen uitvoerbaar is. Zo kan er naar geschreven en van gelezen worden. Waarom is dit een vrij veilige oplossing? De map die algemeen toegankelijk is is niet zichtbaar voor een buitenstaander, zo heeft hij ook geen weet van het bestaan van die map en kan er dus ook geen misbruik van maken. Edoch onze scripts weten wel waar die map zit en kunnen er dus gebruik van maken. Aangezien van onze scripts enkel de resultaten zichtbaar zijn, kan ook op die manier het bestaan van de algemeen toegankelijke map niet achterhaald worden.

Eenmaal de bestanden gekopieërd kan men beginnen met het aanmaken van de database. Eerst wordt er een databank aangemaakt met de naam `eval0x0y`, waarbij `0x0y` het academiejaar is zoals ingegeven in het begin. Daarna wordt in deze databank de verschillende tabellen aangemaakt. Dit gebeurt door eerst een query `CREATE TABLE` op te stellen. De tabel leerkracht ziet er bijvoorbeeld als volgt uit.

```
$stabelleerkracht = "CREATE TABLE leerkracht(
werkneemerID TINYINT(3) UNSIGNED NOT NULL AUTO_INCREMENT,
sectie SET('Muziek','Woord','Dans') CHARACTER SET latin1 COLLATE
latin1_german1_ci NOT NULL,
voornaam VARCHAR(35) CHARACTER SET latin1 COLLATE latin1_german1_
ci NOT NULL,
naam VARCHAR(35) CHARACTER SET latin1 COLLATE latin1_german1_
NOT NULL,
emailadres VARCHAR(70) CHARACTER SET ascii COLLATE ascii_general_
ci NOT NULL,
gebruikersnaam VARCHAR(70) CHARACTER SET ascii COLLATE ascii_gene-
ral_ci NOT NULL,
toegangsrecht SET('A','U') CHARACTER SET latin1 COLLATE latin1_ger-
man1_ci NOT NULL,
wachtwoord VARCHAR(35) CHARACTER SET ascii COLLATE ascii_general_
ci NOT NULL,
PRIMARY KEY (werkneemerID))";
```

Hier worden mooi alle mogelijkheden geïllustreerd. Alle andere tabellen worden op een ge-

lijkaardige manier opgebouwd.

Eenmaal de tabellen bestaan kan men ze invullen. Daartoe worden de CSV bestanden ingelezen. We beginnen met deze voor de graad en de optie. Deze worden iets anders behandeld dan de andere CSV files. Aangezien er geen bewerkingen nodig zijn op deze bestanden, kunnen ze door de database rechtstreeks ingelezen worden in plaats van ze eerst door het script te laten inlezen en dan via een INSERT in de tabel te plaatsen.

```
LOAD DATA LOCAL INFILE 'optie.csv' INTO TABLE optie FIELDS TERMINATED BY ';' LINES TERMINATED BY '\r\n'
```

Dat we moeten specificeren waarmee de lijnen beëindigd worden in de bestanden komt omdat ze binair worden ingelezen en ze anders niet weten waar een lijn eindigt.

Nadien wordt het CSV bestand vak.csv ingelezen. Dit gebeurt in een while lus. Zolang het einde van het bestand niet bereikt is wordt er een lijn ingelezen via de PHP functie fgets. Deze zet de lijn automatisch in een array. Wat makkelijk manipuleerbaar is. Nadien worden de verschillende arraycomponenten via een INSERT in de tabel vak geplaatst. Aangezien we voor het attribuut graad en optie met ID's werken moeten we eerst via een SELECT de ID van de desbetreffende graad en optie achterhalen.

Hetzelfde stramien wordt gevolgd voor de tabel criteria en leerkracht. Bij leerkracht wordt er ook een random wachtwoord gegenereerd op dezelfde wijze als in 4.2.3. Dit wordt ook naar de docent gemaïld.

Voor het invullen van de tabellen cursist, klasgroep, docent en inschrijving moet men eerst enkele bewerkingen doen. Deze tabellen worden namelijk uit één bestand gehaald.

Eerst vult men de klasgroep in. Die moet natuurlijk gelinkt worden aan een graad, optie en een vak. Men achterhaalt de graad door in het CSV bestand de graad te lezen in de vorm ML2. Daarvan leest men de tweede letter. Via een switch conditie weet men dan of men in een lagere, middelbare dan hogere graad zit. Het jaartal haalt men ook uit de graad door gewoon het laatste karakter ervan te nemen. Eenmaal men de graad weet, kan men de gegevens in de tabel klasgroep plaatsen via INSERT. Natuurlijk moet men de ID's van graad, optie en vak achterhalen via een SELECT. Nu wil men geen twee keer dezelfde gegevens in de tabel steken. Dus wordt er eerst gecontroleerd of die reeds aanwezig zijn. Indien dat zo is wordt de INSERT niet uitgevoerd.

Na de klasgroep voert men de gegevens van de tabel docent in. Daartoe moeten de ID's van graad, optie, vak en klasgroep opgevraagd worden. Alsook het werknemernummer.

Als zowel de tuple van klasgroep als docent is ingevuld, kan men de tabel inschrijving invullen. Daarvoor moet men ook de nodige ID's opzoeken via een SELECT. Als laatst vult men dan de tabel cursist in. Dit kan heel eenvoudig via een INSERT wat men moet geen ID's opzoeken.

Bij de laatste tabellen moet men opmerken dat deze samen in een while lus zitten. Men vult

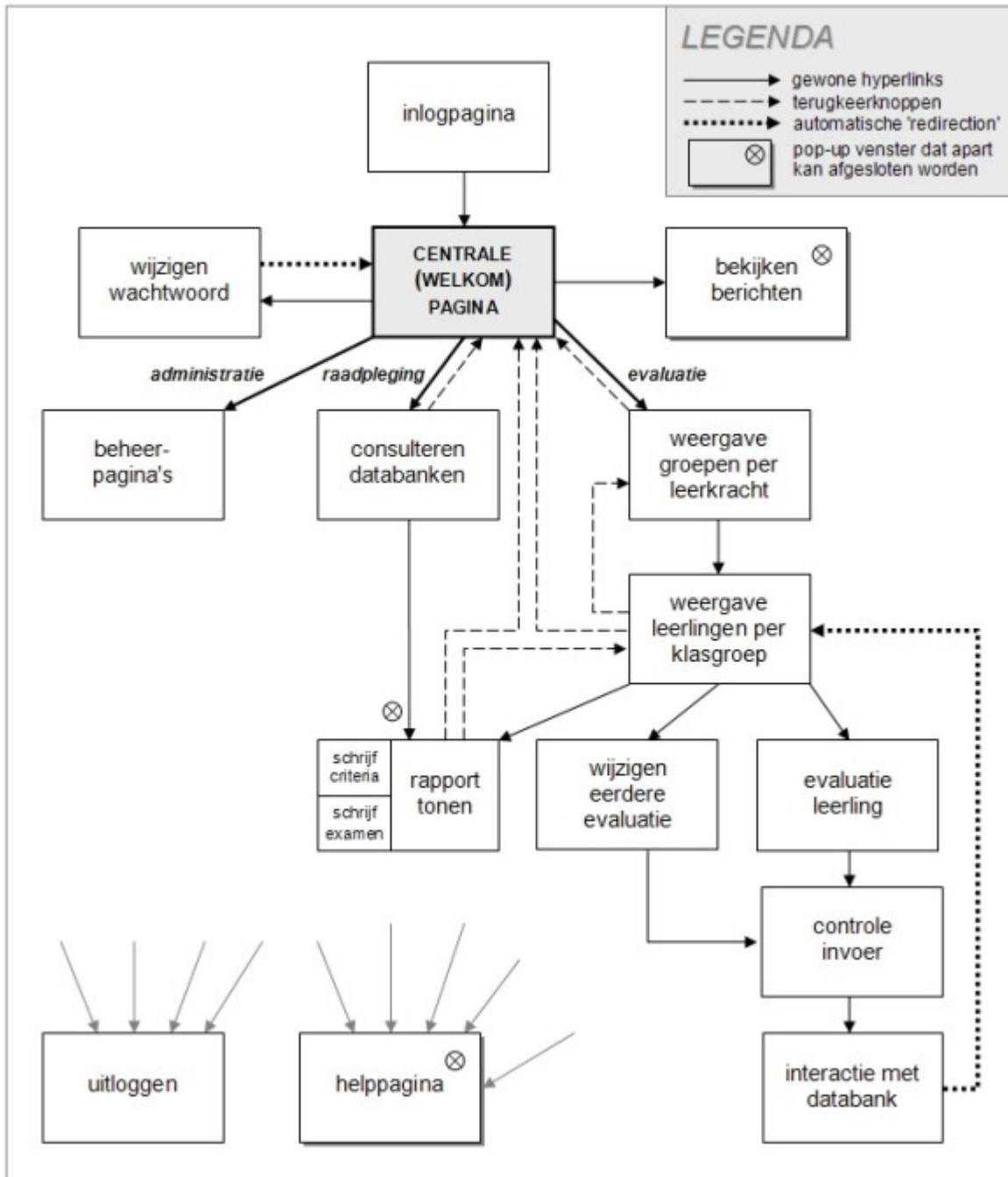
dus eerst een tuple in voor klasgroep, daarna voor docent,..., tot men bij inschrijving is. En zo opnieuw, een tuple voor klasgroep,... tot men het hele bestand muziek.csv doorlopen heeft. Idem voor woord.csv en dans.csv.

Als alles zonder probleem verlopen is krijgt de administrator te zien dat alles goed verlopen is en kan hij terugkeren naar de centrale beheerpagina.

4.3 De nieuwe gebruikerswebsite

4.3.1 Structuur van en navigatie binnen de website

In Figuur 4.7 wordt de structuur van de nieuwe gebruikerswebsite in detail voorgesteld. Op deze figuur wordt elke aangemaakte PHP-pagina weergegeven (16 in het totaal; het vakje ‘beheerpagina’s’ is de link naar de databankbeheerpagina’s die eerder besproken werden en telt dus niet mee; het vakje ‘rapport tonen’ correspondeert met een PHP-bestand met daarin twee geneste PHP-files). Ten opzichte van de oorspronkelijke website werd getracht om de gebruiksvriendelijkheid van het systeem te verhogen, door meer concrete en gestructureerde informatie op het scherm te laten verschijnen en minder gebruik te maken van codes. Er werden enkele nieuwe functies toegevoegd en daarnaast werd ook bijzondere aandacht besteed aan de algemene opmaak: deze werd afgestemd op de weergave van de pagina’s in de meest gebruikte webbrowsers, m.n. Internet Explorer en Mozilla Firefox. De implementatie van de nieuwe website ‘from scratch’ in PHP en HTML vergde ongeveer 3200 lijnen code (CSS exclusief). De oorspronkelijke website (met hier en daar hardgecodeerde PHP-fragmenten) telde circa 3500 lijnen code. Dus ook hier werd — net als bij de databank — meer functionaliteit gecombineerd met een beperktere omvang van de implementatie.



Figuur 4.7: Structuur van de nieuwe gebruikerswebsite. Elk vakje stelt een afzonderlijk PHP-bestand voor. De pijlen geven aan hoe men door de website kan navigeren.

Centraal staat de welkompagina. Dit is de eerste pagina waarop de gebruiker terecht komt na succesvol ingelogd te hebben. Afhankelijk van de toegangsrechten van de gebruiker zal deze kunnen navigeren naar één of meerdere van de drie delen van de website (administratie,

evaluatie en raadpleging). De website is aldus opgevat, dat een gebruiker vanop elke pagina kan uitloggen (behalve vanop de databankinteractiepagina en de pop-up pagina's). Tevens kan de gebruiker op diverse plaatsen helpinformatie opvragen. Om vlot te kunnen navigeren doorheen de webpagina's, werden enkele knoppen voorzien die toelaten om terug te keren naar de vorige pagina of naar de centrale welkompagina. In bepaalde gevallen gebeurt dit automatisch om het gebruiksgemak te verhogen.

De verschillende webpagina's worden hierna besproken. Hierbij zal vooral de functionaliteit van de pagina's aan bod komen (hoe werden de pagina's opgevat? wat zijn de mogelijkheden?). De implementatie in PHP-code zal niet ten gronde besproken worden, aangezien dit minder relevant is.

4.3.2 Authenticatie en andere beveiligingsaspecten

Vooraleer men van het systeem gebruik kan maken, dient men zich eerst in te loggen (dit is de authenticatieprocedure). Hiervoor maakt men gebruik van de toegewezen gebruikersnaam en van het bijhorende wachtwoord. Indien men dit wenst kan men het wachtwoord op regelmatige basis zelf veranderen (via de welkompagina, zie paragraaf 4.3.3). Dit wordt ten zeerste aanbevolen, aangezien het achterhalen van een gebruikersnaam-wachtwoord-koppel een externe persoon in staat stelt om onmiddellijke toegang te hebben tot het systeem. De veiligheid van het systeem ligt dus voor een groot stuk in handen van de gebruikers. Op de inlogpagina (Figuur 4.8) is daarom een link voorzien naar de helppagina's, waar meer uitleg gegeven wordt over hoe men het best een *veilig* wachtwoord uitkiest of samenstelt. Ook kan men via de helppagina's te weten komen wat men moet doen indien men z'n wachtwoord vergeten is of indien men nog geen 'account' heeft en bijgevolg nog niet kan inloggen.

De in het HTML-formulier ingegeven gebruikersnaam en wachtwoord worden via een 'HTTP-request message' van het type 'POST' doorgestuurd naar de webserver. Het is evident waarom hier niet gebruik werd gemaakt van de 'GET'-methode: in dat geval zouden beide gegevens immers zichtbaar in de URL verschijnen van de pagina waarnaar doorverwezen werd. Het prijsgeven van het wachtwoord moet te allen tijde vermeden worden! De server zal vervolgens van het doorgestuurde wachtwoord de MD5-hash berekenen, en dit vergelijken met het MD5-versleutelde wachtwoord, dat uit de databank (zie Tabel 4.1) gehaald wordt door middel van een SQL-bevraging van de tabel 'leerkracht'. Indien deze twee parameters (één berekend, één opgevraagd) identiek zijn, dan wordt de gebruiker door middel van de PHP-*header*-functie doorgestuurd naar de welkompagina. Gaf de gebruiker een verkeerd wachtwoord in (niet overeenkomend met de gebruikersnaam of simpelweg foutief), dan verschijnt een duidelijke foutmelding. De gebruiker kan dan onmiddellijk een nieuwe poging ondernemen, aangezien de invulvelden gewist werden. Het wachtwoord is niet hoofdlettergevoelig (het maakt m.a.w. niet uit of men in kleine of in hoofdletters typt) en wordt bij het intikken aan nieuwsgierige

blikken onttrokken door het als bolletjes of sterretjes (Internet Explorer resp. Mozilla Firefox) te laten verschijnen.

Door de MD5-hash van het wachtwoord te berekenen (met de PHP-*md5*-functie), worden twee potentiële beveiligingsproblemen vermeden. Enerzijds kan het wachtwoord niet meer in zijn niet-versleutelde vorm afgelezen worden uit de databank (\leftrightarrow paragraaf 2.1); anderzijds kan de ingevoerde tekst niet op het systeem uitgevoerd worden. Wanneer men dus scriptcode zou ingeven in het ‘wachtwoord’-invulveld, dan kan dit door de versleuteling niet meer geïnterpreteerd en verwerkt worden door de server.

Het PHP-bestand dat correspondeert met de inlogpagina, nl. ‘login.php’, speelt ook een belangrijke rol bij het configureren van de website bij het begin van een nieuw schooljaar. Wanneer de beheerder de databank én de website in orde wil stellen voor het komende schooljaar, dan dient hij de volgende procedure uit te voeren in de opgegeven volgorde:

- inloggen met z’n persoonlijk *administrator*-wachtwoord
- naar het deel ‘administratie’ gaan
- kiezen voor ‘een nieuwe database maken’
- het jaar specificeren, bijvoorbeeld 0607 voor het schooljaar 2006-2007
- de nodige files uploaden voor het opvullen van de databank
- de vaste items (evaluatiemoment? schooljaar?) van de tabel ‘beheer’ invullen via de databankbeheerpagina’s
- het bestand ‘login.php’ openen met een geschikte editor (bv. *PsPad Freeware Editor*) en in het script op de aangeduide plaats (zie Figuur 4.9) de naam van de net aangemaakte databank invullen (bv. ‘eval0607’). Ook de gebruikersnaam en het wachtwoord die nodig zijn voor de MySQL-databanktoegang moeten op deze plaats ingevuld zijn.

Na deze laatste instelling is de volledige gebruikerswebsite klaar voor het nieuwe schooljaar. Op één van de volgende pagina’s (‘codeperleerkracht.php’) zal dan via een bevraging van de ‘beheer’-tabel het schooljaar en het van tel zijnde evaluatiemoment opgevraagd worden (en in een sessievariabele gestockeerd worden), zodat de evaluaties van de studenten in de juiste ‘evaluatie_X’-tabel terechtkomen ($X = K, P$ of E).

Hierbij aansluitend kunnen we meteen ook de uitlogpagina bespreken. De voornaamste functie van ‘userlogout.php’ is om de sessie die gestart werd bij het inloggen te gaan afsluiten. Concreet houdt dit in dat alle gedefinieerde sessievariabelen (een 9-tal in totaal) bij het bereiken van deze pagina onmiddellijk vernietigd worden. De gebruiker wordt op de hoogte gesteld van het feit dat hij het systeem verlaten heeft. Om te vermijden dat persoon Y, die dezelfde



Figuur 4.8: Inlogpagina van het centraal evaluatie- en rapporteringssysteem van de AMWD Deinze.

computer wil gebruiken waarop persoon X net uitgelogd heeft, met de ‘terug’-knop van de browser zou kunnen terugkeren in het systeem en wijzigingen zou kunnen gaan doorvoeren in naam van persoon X, werden enkele aanpassingen aangebracht in het script. Allereerst werd ervoor gezorgd dat pagina’s niet worden weergegeven indien de sessievariabele ‘username’ (die ingesteld wordt bij het inloggen) niet (meer) ingesteld is (door het feit dat er net uitgelogd werd). Men kan nog wel één pagina terugkeren (op zich geen risico), maar bij de minste muisklik verschijnt een zwart scherm. Daarnaast werd ook een klein stukje JavaScript toegevoegd aan ‘userlogout.php’, dat ervoor zorgt dat de gebruiker aangespoord wordt om het huidige browservenster af te sluiten. In Internet Explorer verschijnt aldus één seconde na het uitloggen een kadertje om de toestemming te vragen het venster te mogen afsluiten, in Mozilla Firefox wordt de gebruikte ‘*self.close()*’-functie echter niet ondersteund.

4.3.3 Welkompagina en generieke look van de website

Toegangsrechten

Zoals eerder vermeld, vormt de welkompagina de centrale uitvalsbasis voor de volledige website. Op deze pagina kan de gebruiker een keuze maken uit één van de volgende opties: evaluatie, raadpleging en administratie. De eerste optie geeft toegang tot het evaluatiegedeelte van de website. In dit deel kunnen leerkrachten hun studenten één voor één evalueren en vervolgens de rapporten bekijken per leerling. Is men echter enkel geïnteresseerd in het rapport van een bepaalde leerling, en was men helemaal niet van plan om leerlingen te gaan evalueren, dan hoeft men niet de volledige website te doorlopen. Men kiest in dat geval ge-

```
// ===== INSTELLING DATABANK =====
// gelieve hier - na het aannemen van de nieuwe databank via de beheerpagina's - de naam van de nieuwe
// databank op te geven a.u.b. Deze dient van de vorm 'eval0x0y' te zijn. Controleer of de aanhalings-
// tekens aanwezig zijn (karakterstring!). U kunt ook de gebruikersnaam en het wachtwoord voor data-
// banktoegang instellen/vijzigen (eveneens karakterstrings, omsloten door aanhalingstekens).

$instelling_DB = 'eval0x0y';
$instelling_username = 'gebruikersnaam';
$instelling_password = 'wachtwoord';

// controleer deze instellingen grondig. Een foutief ingevulde waarde kan het hele systeem lamleggen!
// na instelling zijn alle PHP-scripts klaar voor het volgende schooljaar!
// =====
```

Figuur 4.9: Het stukje code uit 'login.php' waarin de beheerder elk jaar een kleine wijziging moet doorvoeren om de scripts in te stellen voor het nieuwe schooljaar.

woon voor de optie 'raadpleging'. Men kan dan de leerlingen op naam opzoeken en rapporten van de voorbije jaren consulteren. De derde optie biedt een link naar de databankbeheerpagina's. Alles wat het genereren, aanpassen of manipuleren van de databank betreft moet via deze pagina's geregeld worden. Nu is het uiteraard zo dat niet om het even wie de databank zomaar mag aanpassen. Daarom worden drie gebruikerstypes onderscheiden: de administrator, user en visitor. De 'administrator' (beheerder) is de enige persoon die bevoegd is om de administratiepagina's te gebruiken. De persoon in kwestie is dus verantwoordelijk voor het databankbeheer. De leerkrachten zullen het type 'user' toegewezen krijgen. Hiermee kunnen ze evalueren en raadplegen, maar d.m.v. het gebruik van *if*-condities wordt ervoor gezorgd dat de link naar de databankbeheerpagina's onklaar wordt gemaakt. Het derde type, de 'visitor', kan men bijvoorbeeld toewijzen aan die personen die van het systeem gebruik willen maken, maar dan enkel voor het raadplegen van leerlingengegevens en rapporten. We denken hierbij aan de mensen van het secretariaat of aan de directeur. Zij krijgen geen toegang tot evaluatie- of databankbeheerfuncties, wat trouwens ook niet nodig is.

Mededelingen en deadline

Daarnaast kunnen op de welkompagina indien gewenst maximaal drie berichten weergegeven worden. Deze berichten (bv. aankondigingen, opmerkingen betreffende het eerstvolgende rapport, ...) worden door de beheerder via de databankbeheerpagina's opgesteld en opgeslagen in de tabel 'beheer' van de databank. Daarbij geeft de beheerder eerst een titel op, en vervolgens het bericht. Indien bijvoorbeeld het titelveld van de eerste mededeling in de 'beheer'-tabel is ingevuld, dan wordt dit automatisch gedetecteerd door het script van de welkompagina. Via een query wordt deze titel dan opgevraagd, en weergegeven rechtsonderaan het scherm, in het kadertje 'berichten'. Klikt men op deze titel, dan komt men op de berichtpagina terecht. Vanuit het PHP-bestand 'berichten.php' wordt immers het met de titel corresponderende bericht opgevraagd, en samen met de titel weergegeven op deze pop-up

pagina (men kan dit venster sluiten zonder daarom de volledige website te moeten afsluiten). Analoog kunnen tot drie mededelingen tegelijk worden weergegeven. Een aparte ‘entry’ werd in de ‘beheer’-tabel voorzien voor een deadline. Deze deadline, die bijvoorbeeld de uiterste datum voorstelt waarop leerlingen nog kunnen geëvalueerd worden via het systeem, wordt dan onder de berichten in hetzelfde kadertje weergegeven. Zijn er geen mededelingen of deadlines opgegeven, dan is er ook geen kadertje te zien.

Wachtwoord wijzigen

Via de link naar ‘wizigwachtwoord.php’ kunnen gebruikers ook hun wachtwoord wijzigen. Daartoe dient men eerst het oude wachtwoord in te tikken, en vervolgens het nieuwe twee maal (ter controle op typfouten). Indien het oude wachtwoord correspondeert met de ingelogde gebruiker en indien tweemaal dezelfde karakterstring (enkel ASCII!) wordt opgegeven, dan wordt via de SQL-*update*-procedure het nieuwe MD5-versleutelde wachtwoord opgeslagen in de tabel ‘leerkracht’. Na het bevestigen verschijnt ofwel een foutmelding, ofwel de melding dat het wachtwoord gewijzigd werd. In het laatste geval wordt de gebruiker automatisch teruggestuurd naar de welkompagina. Terzijde kan nog vermeld worden dat er ook een link voorzien is naar het e-mailadres van de beheerder; indien men hierop klikt wordt automatisch de mail-client van de gebruiker opgestart (typisch Outlook), zodat een e-mail met vragen of suggesties kan verstuurd worden.

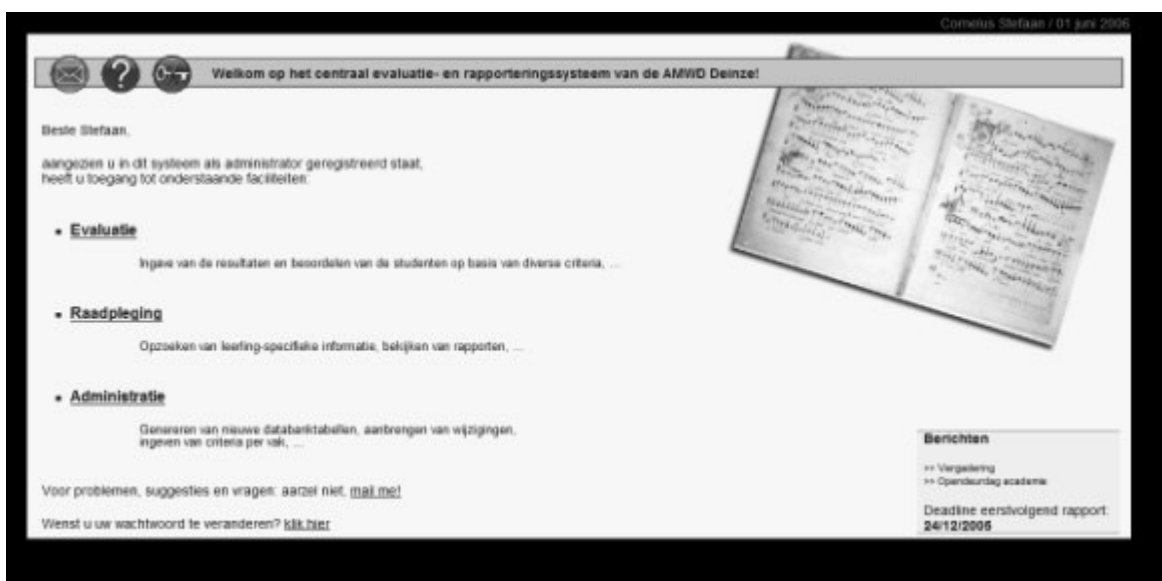
Generieke look

Vanaf de welkompagina wordt ook de ‘look’ van de website duidelijk. Er werd getracht van een vertrouwde, gelijkaardige structuur te steken qua opmaak in alle webpagina’s van de gebruikerswebsite. Deze ‘generieke look’ wordt gecreëerd door vier elementen:

- enkele pagina’s niet te na gesproken, komen er linksbovenaan telkens drie knoppen voor. Deze laten de gebruiker toe om een mail te sturen naar de beheerder (het envelopje), om de helppagina horende bij de bewuste webpagina te raadplegen (het vraagteken) en om uit te loggen (de doorkruiste sleutel).
- op elke pagina komt bovenaan een titelbalk voor. Deze balk bevat informatie over waar men zich in het systeem bevindt, bv. ‘*Overzicht van de klasgroepen*’ of ‘*Rapport van Jan Peeters*’.
- rechtsbovenaan wordt de naam van de ingelogde gebruiker weergegeven, samen met de datum. Waar mogelijk wordt de gebruiker ook met de voornaam aangesproken.
- tot slot spelen ook de kleurschema’s een belangrijke rol. Omwille van het esthetische, maar ook om de samenhang te beklemtonen, werd gewerkt met een relatief beperkt

aantal kleuren (zwart, bordeaux, grijs, wit en oranje). De helppagina's daarentegen hebben steeds een lichtblauwe achtergrond.

De opmaak werd geïmplementeerd d.m.v. een Cascading Style Sheet. Dit CSS-bestand projecteert zijn opmaak op alle pagina's die dit opmaakschema gebruiken. Bijzondere aandacht werd besteed aan de verschillen in weergave tussen Internet Explorer en Mozilla Firefox. Qua opmaak werd gezocht naar de grootst gemene deler, een opmaakschema dat op beide browsers tot z'n recht komt. Dit is minder evident dan het lijkt, aangezien Mozilla Firefox de CSS-opmaak strikt implementeert, terwijl Internet Explorer — zoals bij HTML — de regels meestal vrij interpreteert (lees: met de voeten treedt!).



Figuur 4.10: De welkompagina van de gebruikerswebsite. Van hieruit kunnen een groot aantal acties ondernomen worden. Vanuit andere pagina's wordt dan ook naar deze pagina teruggekeerd. Merk de generieke opmaak van de webpagina op.

4.3.4 Weergave van de klasgroepen en van de studenten

Klasgroepen per leerkracht

Wanneer de gebruiker kiest voor de optie 'evaluatie', dan komt hij/zij allereerst terecht op de pagina 'codeperleerkracht.php'. Hier wordt in het PHP-script een bevraging uitgevoerd van de 'beheer'-databanktabel. Daartoe moet eerst een connectie gemaakt worden met de databank. De daarvoor noodzakelijke gegevens (naam databank, gebruikersnaam en wachtwoord) werden eerder (op de inlogpagina) reeds in sessievariabelen gestockeerd en kunnen dus makkelijk opgeroepen worden. Uit de 'beheer'-tabel worden het huidige schooljaar en evaluatiemoment opgevraagd. Deze parameters, die bepalen welke criteria er dienen beoor-

deeld te worden en waar de evaluatiegegevens in de databank moeten terechtkomen, worden in sessievariabelen opgeslagen, zodat deze databankbevraging niet op elke volgende pagina moet herhaald worden. Ze worden ook op het scherm weergegeven.

Via de sessievariabelen die ingesteld werden op de inlogpagina kan nagegaan worden welke gebruiker ingelogd is (naam, werknemerID, toegangsrecht, ...). Alle klasgroepen waaraan deze ingelogde leerkracht les geeft zullen via een complexe sequentie van databankbevragingen opgezocht worden en — in vergelijking met het vroegere systeem — op een meer gestructureerde wijze weergegeven worden. Dit betekent dat men niet meer één (lange) lijst krijgt met klasgroepen en hun bijhorende, nogal cryptische lettercodes, maar eerder een opdeling van de klasgroepen per vak, afdeling en graad, met weergave van het aantal leerlingen per klasgroep en het aantal leerlingen dat reeds geëvalueerd werd voor het huidige evaluatiemoment. Wanneer alle leerlingen van een bepaalde groep geëvalueerd zijn, dan verschijnt rechts van de groepinformatie een melding in het groen ‘alles ingevuld!’. Zo heeft de leerkracht direct een duidelijk overzicht van welke groepen hij nog dient te evalueren. De complexiteit van de bevragingen is voornamelijk te wijten aan het feit dat er ook gemengde groepen (met leerlingen uit verschillende graden) kunnen voorkomen (de MIX-groepen, zie Tabel 4.2). Aangezien er voor de weergave ook een sortering moet zijn van de groepen per graad, dienen deze MIX-groepen (die niet aan een aparte graad kunnen toegewezen worden) dus eerst afgezonderd te worden van de andere groepen. Daarna worden ze achter de ‘normale’ groepen weergegeven. Verder is het zo dat men ook de klasgroepen van andere leerkrachten moet kunnen raadplegen (dit was in het oorspronkelijke systeem eveneens mogelijk) door een naam te selecteren uit het keuzeveld bovenaan de pagina. Wanneer men de naam van een leerkracht aanklikt, dan verschijnt eerst een lijstje van alle vakken die door deze leerkracht gegeven worden, opnieuw gevolgd door de gedetailleerde weergave van diens klasgroepen per vak, afdeling en graad.

We willen u de details omtrent de implementatie in PHP-code besparen, maar geven in de plaats daarvan in pseudocode de hiërarchie weer die terug te vinden is in de vele controlestructuren die verantwoordelijk zijn voor de correcte weergave van de klasgroepen:

```

for(elk van de vakken die gegeven worden door de ingelogde of geselecteerde leerkracht) {
    print de vaknaam af met opmaak X

    $query_statement = "SELECT afdeling FROM ... inner join ... using(...) WHERE ...";
    (opzoeken van de afdelingen waar dat vak gegeven wordt door de leerkracht in kwestie)
    $query_uitvoering = mysql_query($query_statement,$connectie) or (die(mysql_error()));
    $query_resultaat = mysql_num_rows($query_uitvoering);

    for(elk van de afdelingen waarop dit vak door deze leerkracht gegeven wordt) {
        - print de afdelingnaam af met opmaak Y
    }
}

```

(analoog als hierboven) opzoeken van alle leerlingen die dit vak bij deze leerkracht in deze afdeling volgen, en opsplitsen door middel van subqueries in de leerlingen van normale groepen en de leerlingen van MIX-groepen

```

for(de gewone groepen) {
    - print de graadnaam af met opmaak Z

    - geef de groepen weer met leerlingen die in deze graad, bij deze leerkracht, het gegeven vak volgen in deze afdeling
    - zoek ook het aantal (geëvalueerde) leerlingen per groep op en geef deze getallen weer
    - controleer of de melding 'alles ingevuld!' moet weergegeven worden
}
for(de MIX-groepen) {
    - print 'groepen met lln. uit meerdere graden' af met opmaak Z

    - geef de groepen weer met leerlingen die in deze gemengde groep, bij deze leerkracht, het gegeven vak volgen in deze afdeling
    - zoek het aantal (geëvalueerde) leerlingen per groep op en geef deze getallen weer
    - controleer of de melding 'alles ingevuld!' moet weergegeven worden
}
}
}

```

Het uiteindelijke resultaat wordt weergegeven in Figuur 4.11. Net onder de naam van de leerkracht wordt ook het totaal aantal groepen en het totaal aantal leerlingen van die leerkracht weergegeven. Het feit dat deze cijfers vóór de weergave van de klasgroepen, en dus niet onderaan de pagina weergegeven moesten worden, betekende dat hier van enkele kunstgrepen gebruik moest gemaakt worden om deze cijfers 'a priori' te berekenen.

Studenten per klasgroep

Bij het aanklikken van een groepsnaam op de net besproken pagina, wordt de gebruiker doorgestuurd naar 'codeperklasgroep.php'. Door middel van de 'GET'-methode worden hierbij de werknemerID, graadID, vakID en groepID doorgegeven. Klikt men echter op een MIX-groep, dan zal de doorgestuurde GET-variabele \$_GET['gra'] (die normaal de graadID bevat) de string 'mix' als waarde hebben. Gebruik makende van deze identifiers kan dan op de 'codeperklasgroep'-pagina een aantal queries uitgevoerd worden, zodanig dat alle kenmerken die de leerlingen uit deze groep gemeenschappelijk hebben (leerkracht, vak, graad, groep, afdeling) als gewone tekst bovenaan het scherm kunnen weergegeven worden. Ook worden

The screenshot shows a web interface for 'BACKELAND GUDRUN' with 9 classes and 25 students. It displays three subject areas with their respective class groups and student counts.

ALGEMENE MUZIEKCULTUUR			
Deinze			
muziek middelbare graad 1			
groep.A	18 leerlingen	5 ingevuld	---
groep.B	1 leerling	0 ingevuld	---
muziek middelbare graad 2			
groep.B	1 leerling	1 ingevuld	alles ingevuld!
muziek middelbare graad 3			
groep.V	2 leerlingen	0 ingevuld	---
ALGEMENE MUZIKALE VORMING			
Deinze			
muziek lagere graad 1			
groep.O	1 leerling	0 ingevuld	---
muziek lagere graad 2			
groep.A	1 leerling	1 ingevuld	alles ingevuld!
MUZIEKGESCHIEDENIS			
Deinze			

Figuur 4.11: Weergave van de klasgroepen van de ingelogde of de geselecteerde leerkracht.

aan de hand van deze identifiers de namen van de leerlingen opgezocht, de optie die ze volgen (groepen kunnen leerlingen uit verschillende opties bevatten) en wordt gecontroleerd of deze leerlingen reeds beoordeeld werden rond Kerst, Pasen en het einde van het jaar (door na te trekken of de tabel 'evaluatie_X' reeds data bevat voor een leerling met een bepaald stamnummer die een bepaald vak met een zekere vakID volgt in de groep gekarakteriseerd door groepID, dit voor X= K, P en E). Al deze informatie wordt dan gestructureerd weergegeven in een tabel (zie Figuur 4.12).

Naast iedere rij zijn telkens twee knoppen voorzien. De ene knop, met het opschrift 'vul in', laat toe van de leerling in die rij te gaan evalueren. De knop licht groen op wanneer er met de muisaanwijzer over het knopvlak 'gehoverd' wordt. Deze knop is niets anders dan een met CSS opgemaakte hyperlink, die toegang geeft tot 'evaluatieleerling.php'. Is de leerling voor het geldig zijnde evaluatiemoment echter al geëvalueerd, dan wijzigt de linkse knop in een 'wijzig'-knop (licht oranje op), die verbinding geeft met 'wijzigevaluatie.php'. De rechtse knop (blauw oplichtend) laat toe om het rapport van die leerling te bekijken ('overzichtleerling.php'). Het rapport bevat niet enkel de evaluatie door de ingelogde leerkracht, maar ook alle cijfers en evaluaties die reeds ingegeven werden door andere leerkrachten voor de voorbije evaluatiemomenten van het huidige schooljaar. Via de knop onderaan de pagina kan men makkelijk terugkeren naar de klasgroepen van de geselecteerde of de ingelogde leerkracht.

The screenshot shows a web interface for 'BACKELAND GUDRUN'. At the top, it says 'Overzicht van de leerlingen per klasgroep' and 'Schooljaar 2006-2006 | Evaluatiemoment: Kerst'. Below this, there is a breadcrumb trail: 'algemene muziekcultuur' > 'muziek middelbare graad 1' > 'Dansza, klas A -> 15 leerlingen'. A table lists 15 students with columns for 'Optie', 'Naam student', 'Kerst', 'Pasen', 'Eind', and two columns of status indicators (e.g., 'vol in', 'afgesloten').

Optie	Naam student	Kerst	Pasen	Eind		
samenspel	Cleymaet Katrine	ingevuld	ingevuld	---	vol in	afgesloten
samenspel	Coene Rebecca	ingevuld	---	---	vol in	afgesloten
instrument	De Rijcke Thijs	---	---	---	vol in	afgesloten
samenspel	De Wachter Tim	ingevuld	---	---	vol in	afgesloten
instrument	Laegens Lenore	---	---	---	vol in	afgesloten
samenspel	Maeyens Stan	ingevuld	---	---	vol in	afgesloten
instrument	Marysse Chloé	---	---	---	vol in	afgesloten
instrument	Onderbeke Mattis	---	---	---	vol in	afgesloten
samenspel	Schelewaete Simon	---	---	---	vol in	afgesloten
samenspel	Van Aubréve Nick	---	---	---	vol in	afgesloten
samenspel	Van Cauwenbergh Sara	---	---	---	vol in	afgesloten
samenspel	Van De Geuchte Delphine	ingevuld	---	---	vol in	afgesloten
samenspel	Vande Kerckhove Lisa	---	---	---	vol in	afgesloten
instrument	Van Wassenhove Edith	---	---	---	vol in	afgesloten
instrument	Voet Judith	---	---	---	vol in	afgesloten

Figuur 4.12: Weergave van de studenten in de gekozen klasgroep.

4.3.5 Invoerrechten en -pagina

Wanneer men op de ‘vul in’-knop klikt, dan worden opnieuw een aantal essentiële identifiers doorgestuurd via een ‘HTTP-request message’ van het type ‘GET’ naar de invoerpagina, ‘evaluatieleerling.php’. Daar wordt eerst gecontroleerd of de via GET ontvangen ‘werknemerID’ overeenkomt met de ‘userID’ die opgeslagen zit in een sessievariabele. Er wordt met andere woorden gecontroleerd of de ingelogde gebruiker wel gemachtigd is om deze leerling te evalueren. Indien dit niet het geval is, wordt de gebruiker erop attent gemaakt dat hij de gegevens van deze leerling enkel kan bekijken, maar dat hij niet mag evalueren. In het andere geval verschijnt de invoerpagina met de criteria die voor die trimester opgegeven werden. Verder wordt door middel van enkele databankbevestigingen nagegaan of er al dan niet een (partieel) examen voorzien is en of er met één totaal of met een opsplitsing in categorieën gewerkt wordt. Afhankelijk van het evaluatiemoment en van de sectie, zullen de titels ook aangepast worden (bv. ‘rapport Kerst’ voor de secties muziek en dans, maar ‘Evolutierapport 1’ voor de sectie woord). Ook alle administratieve gegevens van de gekozen student (uit de tabel ‘cursist’) worden bovenaan de pagina weergegeven.

Ter illustratie hernemen we het voorbeeld uit de Tabellen 4.3 en 4.5, voor het evaluatiemoment ‘Pasen’. Tabel 4.5 gaf aan hoe de evaluatiegegevens terecht kwamen in de databanktabel ‘evaluatie_P’. In Figuur 4.13 is te zien hoe de invoerpagina die aanleiding gaf tot deze gegevens, eruit ziet. Er is een duidelijke opdeling in de te beoordelen criteria bovenaan (in dit

geval enkel met cijfer- en tekstvelden, vanwege de instellingen in Tabel 4.3) en de beoordeling van het partiële examen onderaan. Aangezien de leerkracht gekozen had voor een O-type beoordeling, met als categorieën ‘dagelijks werk’ en ‘theorie’, worden onderaan twee invulvelden weergegeven (samen met het maximum waarop beoordeeld dient te worden). Wanneer de leerkracht vervolgens deze velden in de juiste volgorde invult, dan zal bij het verlaten van het veld ‘theorie’ (door erbuiten te klikken of de TAB-toets te gebruiken) automatisch het totaal berekend en weergegeven worden. Deze client-side berekening werd geïmplementeerd d.m.v. een eenvoudig stukje innerHTML JavaScript. Men moet het totaal dus niet zelf berekenen. Indien men op deze pagina op de helpknop klikt, dan krijgt men de specifieke helppagina (die uitleg geeft over hoe de verschillende types velden — zie pagina 4.1.1 — dienen ingevuld te worden) te zien (Figuur 4.14).

The screenshot shows a web application interface for evaluating a student. The page is titled "Evaluatie van Aernoudts Ann-Lara" and includes a navigation bar with "Schooljaar 2005-2006" and "Evaluatiemoment: Pasen".

AERNOUDTS ANN-LARA
— muziek lagere graad 1
— optie 'algemene muzikaleer'
— **ALGEMENE MUZIKALE VORMING**
— Deinze, klas D

gesl.: V geb.datum: 18/05/1997 (8 jaar)
Zomerstraat 37
9800 Deinze (België)
tel 1: 09/280 30 65 tel 2: 0477/34 85 67

TE BEOORDELEN CRITERIA (RAPPORT PASEN)

Gelevez de hierna volgende criteria te beoordelen. (zie 'help' voor meer informatie)
U kunt makkelijk van het ene naar het andere veld springen m.b.v. de TAB-toets...

algemene kennis	<input type="text" value="17"/> op 20
gehoor	<input type="text" value="15.5"/> op 20
houding	<input type="text" value="Werk goed mee"/> op 20
juiste toon zingen	<input type="text" value="Blijven oefenen!"/> op 20
lezen van de noten	<input type="text" value="Geef rief"/> op 20

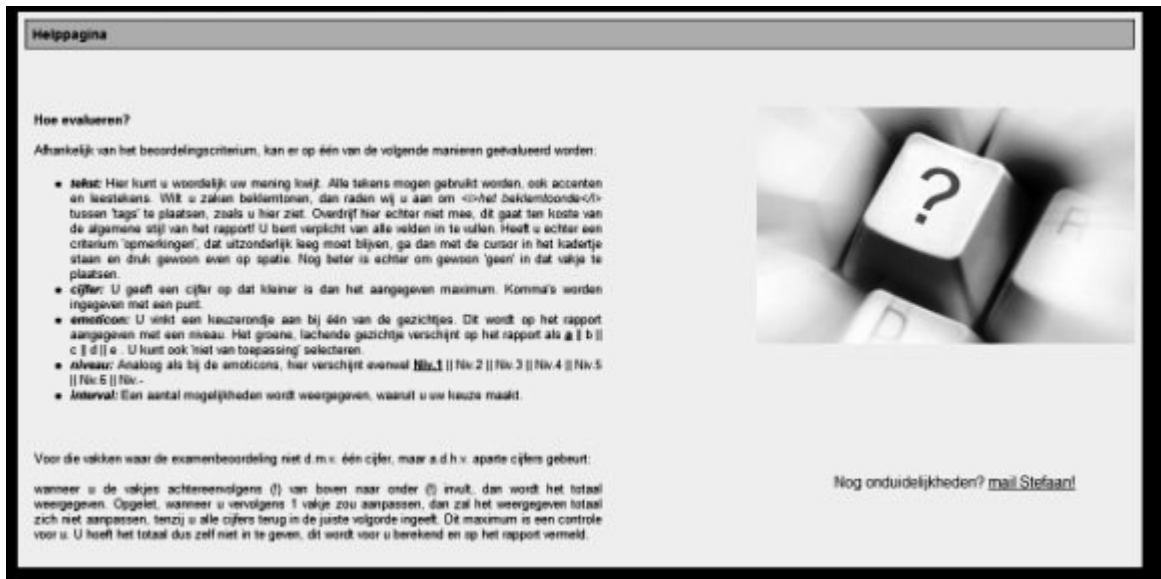
PARTEEL EXAMEN PASEN:

Gelevez de hiernavolgende criteria afzonderlijk te quoteren. Het totaal wordt voor u berekend.
(Wanneer u de velden achtereenvolgens én in de juiste volgorde invult, zal bij het verlaten van het laatste veld het totaal weergegeven worden)

dagelijks werk	<input type="text" value="26"/> op 30
theorie	<input type="text" value="54"/> op 70

Het totaal bedraagt: 80 op 100

Figuur 4.13: Invulvelden voor de te beoordelen criteria en examenresultaten.



Figuur 4.14: Helppagina als ondersteuning bij het invullen van de velden.

4.3.6 Controle van de input en invoer in de databank

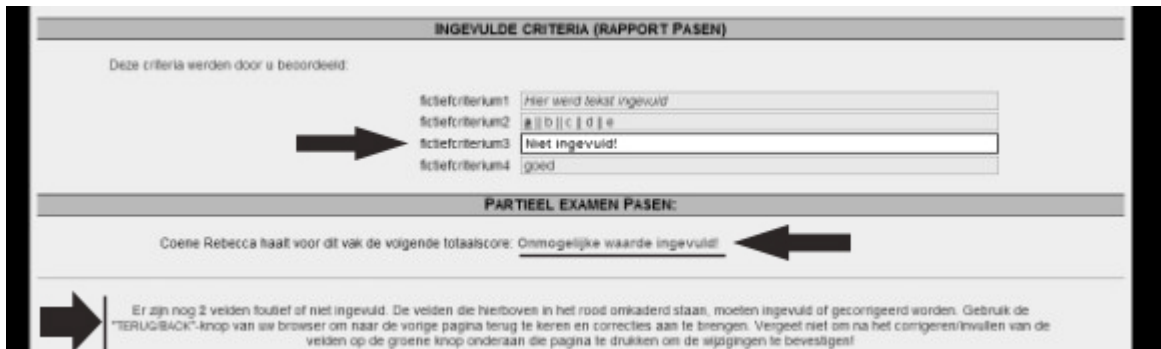
Inputcontrole

De pagina 'controleinvoer.php' doet dienst als een buffer tussen de invoer- en de databankinteractiepagina. De bedoeling van deze pagina is te vermijden dat er foutieve gegevens in de databank zouden terechtkomen of dat er velden niet zouden ingevuld zijn. Daarnaast biedt het de leerkracht de kans om nog eens te overlezen wat hij/zij zonet heeft ingevuld. Het PHP-script van deze pagina is vrij complex, omdat nu niet alleen de juiste criteria moeten opgezocht en weergegeven worden, maar ook de zonet ingevulde waarden; daarbovenop moet het script controleren of er velden leeg gebleven zijn en of de ingevulde getallen niet groter zijn dan het toegelaten maximum of kleiner dan nul.

De ingevulde waarden worden vanaf 'evaluatieleerling.php' doorgestuurd via de POST-methode. Elke POST-parameter krijgt dezelfde naam als het corresponderende beoordeelde criterium, waaruit de spaties verwijderd werden. De beoordeling van het criterium 'lezen van noten' (nl. 'gaat vlot', zie Tabel 4.5) wordt dus doorgegeven als: `$_POST['lezenvannoten']='gaat vlot'`. Wanneer er bij het invullen een fout blijkt opgetreden te zijn, dan zal niet de ingevulde waarde weergegeven worden, maar dan wordt het vakje naast het criterium rood omkaderd en wordt er een foutmelding in weergegeven. Al naargelang verschijnt er dus:

- 'Niet ingevuld!' bij een vergeten veld...
- 'Onmogelijke waarde ingevuld!' indien men bijvoorbeeld -2/10 of 11/10 ingaf...

Onderaan de pagina verschijnt nog eens het aantal velden dat nog niet of foutief ingevuld is, samen met enkele richtlijnen omtrent wat de gebruiker te doen staat. Deze zal namelijk gevraagd worden om met de ‘terug’-knop van de browser naar de vorige pagina terug te keren en de corrupte velden te corrigeren of in te vullen. In Figuur 4.15 wordt een voorbeeld gegeven, waarbij een criteriumveld niet ingevuld werd en het examen beoordeeld werd met een onmogelijk cijfer. De pijlen geven de foutmeldingen aan.



Figuur 4.15: Het systeem heeft gedetecteerd dat er velden niet of foutief ingevuld werden en wijst de gebruiker hierop. De pijlen geven de foutmeldingen aan.

Invoer in de databank

Als en slechts als alle velden correct ingevuld zijn, verschijnt onderaan de pagina een knop met het opschrift ‘Voeg evaluatie <voornaam> <naam> toe aan de databank’. Deze link brengt de gebruiker op de pagina ‘interactiedb.php’. Op deze pagina kan de gebruiker niets concreet aanvangen. De enige maar wel uitermate belangrijke functie van deze pagina is de ingevulde gegevens op de juiste plaats én op de juiste manier te stockeren in de databank. Vervolgens verschijnt een kort bericht dat de gebruiker op de hoogte stelt van het succes van de databankinteractie. Indien alles goed verloopt, dan krijgt men de melding dat ‘de evaluatie van <voornaam> <naam> aan de databank werd toegevoegd’. Gaat er iets fout, dan wordt door middel van de *mysql_error*-functie een foutbericht met verdere informatie (gegenereerd door het databankbeheersysteem en bijgevolg in het Engels) op het scherm weergegeven.

In het script worden een vrij groot aantal SQL-manipulaties (van het type ‘SELECT’ en ‘INSERT’) uitgevoerd. De ‘SELECT’-queries zijn hoofdzakelijk bedoeld om in de databank-tabel ‘criteria’ na te gaan welke criteria er dienen beoordeeld te worden voor het geldige evaluatiemoment. Ook wordt daarmee nagegaan hoe de examens georganiseerd zijn en welke cijfers er dus nodig zijn in de databank. Met de ‘INSERT’-opdrachten wordt de tabel ‘evaluatie_X’ dan opgevuld zoals eerder uitgelegd werd in paragraaf 4.1.2. Om die tabel juist te kunnen invullen, moeten de beoordelingscriteria gekend zijn. Vandaar dat de criteria dus opnieuw moeten opgevraagd worden. Zo ontstaat in het script een geschrinkt systeem van

if-then-else-lussen, ‘SELECT’- en ‘INSERT’-opdrachten, die door hun onderlinge samenwerking ervoor zorgen dat de juiste waarden in de juiste kolommen van de tabel ‘evaluatieX’ terechtkomen. Aangezien de wijze waarop de tabel wordt opgevuld reeds eerder uit de doeken werd gedaan, komen we er hier niet verder op terug. In het geval van een examenbeoordeling d.m.v. een aantal categorieën (type ‘O’), wordt op de controlepagina het totaal berekend. Men moet dus onderscheid maken tussen het JavaScript uit de *evaluatie*pagina (dat het totaal enkel ter controle berekende) en de effectieve berekening van het totaal op de *controle*pagina. Het resultaat van deze laatste bewerking wordt net zoals de andere gegevens doorgegeven via een sessievariabele aan de databankinteractiepagina.

Nadat de interactie met de databank zich voltrokken heeft en dus enkele seconden na het verschijnen van de bevestiging of de foutmelding, wordt de gebruiker automatisch omgeleid naar de pagina met de weergave van de leerlingen per klasgroep. Deze automatische ‘redirection’ gebeurt omwille van twee redenen:

- enerzijds om te vermijden dat de gebruiker op de ‘terug’-knop van zijn browser zou klikken, wat als consequentie zou kunnen hebben dat een zelfde databankinteractie twee keer zou worden uitgevoerd (wat aanleiding geeft tot foutmeldingen wegens het niet respecteren van het relationele databankmodel: dubbele tupels!)
- anderzijds voor het gebruiksgemak: de leerkracht beoordeelt een leerling, voegt de evaluatie toe aan de databank en wordt vanzelf weer teruggestuurd zodat hij/zij onmiddellijk de volgende leerling voor evaluatie kan selecteren

Bij het ‘redirecten’ naar ‘codeperklasgroep.php’ worden de gegevens van de beoordeelde student (allerlei identifiers) meegegeven via de URL (dus GET-methode). Hierdoor zal in de tabel die te zien is op Figuur 4.12, het label ‘ingevuld’ verschijnen bij de student in kwestie en zal de knop ‘vul in’ omgevormd worden tot de knop ‘wijzig’.

4.3.7 Wijzigen of bekijken van de gegevens per student

Aanpassen van eerdere evaluaties

Het wijzigen van een eerdere evaluatie is in het nieuwe systeem mogelijk, op voorwaarde dat men zich nog steeds in hetzelfde evaluatiemoment bevindt. Er zal hier niet verder ingegaan worden op de code achter de ‘wijzigevaluatie.php’-pagina: deze kan het best omschreven worden als een amalgaam van de code van ‘evaluatieleerling.php’ en ‘controleinvoer.php’. Na het uitvoeren van de correcties, wordt er opnieuw een inputcontrole uitgevoerd alvorens de gegevens in de databank op te slaan. Om inconsistenties, gecreëerd door het aanbrengen van wijzigingen, te vermijden, wordt op het moment dat ‘wijzigevaluatie.php’ wordt opgeroepen, de te wijzigen evaluatie volledig gewist uit de databank. Nadat de gebruiker wijzigingen heeft aangebracht, dient deze zeker op de knop onderaan de pagina te klikken ter bevestiging.

Aldus wordt de mix van oude en gewijzigde gegevens dan terug in de databank opgenomen. Bevestigt men niet, dan zal de leerling als ‘niet geëvalueerd’ beschouwd worden door het systeem (dit is ook effectief zo, aangezien er geen gegevens voor die student voor dat vak in de databanktabel ‘evaluatie_X’ voorkomen). Zowel bij het opslaan als bij het wijzigen van evaluaties worden de gegevens dus telkens in blokken weggeschreven naar de databank (‘alles of niets’-principe om inconsistenties te voorkomen).

Weergave van de rapporten

Tijdens en na de evaluatie van een leerling door één of meerdere leerkrachten, kan men het rapport van deze leerling opvragen. De pagina ‘overzichtleerling.php’ collecteert alle op dat moment beschikbare gegevens, voor alle vakken die door die student gevolgd worden en dit voor alle gepasseerde evaluatiemomenten. Aangezien deze pagina verschillende databanktabellen (‘cursist’, ‘criteria’, ‘evaluatie_X’, ‘leerkracht’, ...) dient te bevragen om alle noodzakelijke gegevens te bundelen, is het één van de meest complexe scripts. Er werden dan ook twee stukken code afgezonderd, die meerdere keren dienen doorlopen te worden (nl. 1 maal voor elk evaluatiemoment). Hiervan werden twee aparte PHP-bestanden gemaakt, ‘schrijfcriteria.php’ en ‘schrijfexamen.php’, die dan in het masterbestand ingesloten werden d.m.v. de PHP-*include*-functie. Deze functie genoot de voorkeur op PHP-*require()*, omdat deze laatste methode — aldus de PHP-documentatie — aanleiding kan geven tot fatale fouten.

Bovenaan de pagina worden opnieuw de administratieve gegevens van de leerling in kwestie getoond (cfr. Figuur 4.16). Behalve het adres, de telefoonnummers en het geslacht wordt ook de leeftijd weergegeven (deze wordt berekend uit de geboortedatum). Daaronder verandert de lay-out van de pagina in een kolommenstructuur. De twee buitenste kolommen werden van een gelijkaardige opmaak voorzien d.m.v. CSS; de middelste kolom diende voldoende te contrasteren om de leesbaarheid te verhogen. De hoofdingen boven de drie kolommen, die aangeven om welk rapport het gaat, worden ook hier aangepast in functie van de sectie en het evaluatiemoment (zie eerder). Daaronder wordt voor elke combinatie van een graad en een optie waarin de student vakken volgt, een titel weergegeven. De ordening is dusdanig dat de vakken uit de hogere graden eerst komen, gevolgd door de eventuele vakken die de student volgt in lagere graden. Onder elke titel verschijnt dan per vak een massief blok, eveneens opgedeeld in kolommen. De concrete inhoud van de drie kolommen (beoordeling van de criteria) wordt samengesteld door een aantal geneste for- en if-lussen in het script te doorlopen (opzoeken van de geëvalueerde criteria voor dat vak, binnen die graad, enz.). Hier wordt gebruik gemaakt van ‘schrijfcriteria.php’ om de (al dan niet gecategoriseerde) criteria (zie Figuur 4.17) uit te schrijven. Dan volgt — nog steeds binnen hetzelfde blok — een kleine scheidingstitel, om het onderscheid te maken met de evaluatie van de examens. Hier komt ‘schrijfexamen.php’ zijn rol vervullen. De examenresultaten worden (in het geval van opdeling in categorieën) gestructureerd weergegeven, alsook het berekende totaal. Er zijn

knoppen voorzien om vanop deze pagina terug te keren naar de centrale welkompagina, of naar de weergave van de studenten per klasgroep (zoals aangeduid op Figuur 4.7).

Rapport van Coene Rebecca

SAMWD DEINZE - RAPPORT

COENE REBECCA - SECTIE MUZIEK

gest. V geb datum: 29/01/1995 (13 jaar)

Melstraat 3
9650 NEVELE (België)

tel.1: 09266 01 50 tel.2: 0478/40 23 46

RAPPORT KERST **RAPPORT PASEN** **RAPPORT EIND**

MIDDELBARE GRAAD 1 - OPTIE SAMENSPEL

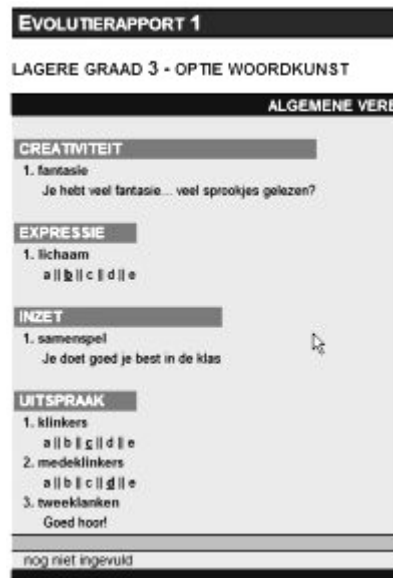
ALGEMENE MUZEKCULTUUR, DEINZE, KLAS A (GUDRUN BACKELAND)		
1. Sctielcriterium1 teetje123	1. Sctielcriterium1 teet	nog niet ingevuld
2. Sctielcriterium2 a b c d e	2. Sctielcriterium2 a b c d e	
3. Sctielcriterium3 a b c d e	3. Sctielcriterium3 a b g d e	
4. Sctielcriterium4 voldoende	4. Sctielcriterium4 goed - zeer goed	
EXAMENS		
geen parteel examens	Totaal behaakt: 82/100	nog niet ingevuld

terug naar de klasgroep

terug naar de hoofdpagina

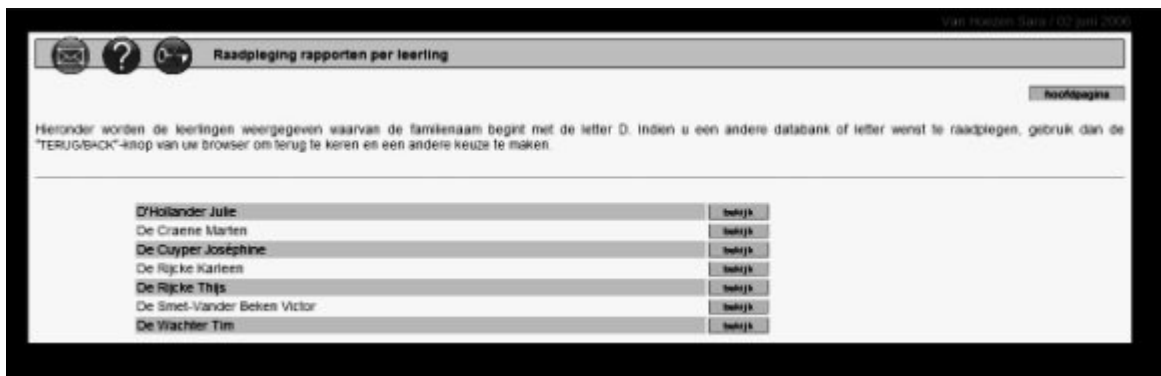
Figuur 4.16: Voorbeeld van de weergave van een eenvoudig rapport.

Zoals eerder vermeld, kunnen de rapporten ook rechtstreeks opgevraagd worden vanuit het ‘raadpleeg’-gedeelte van de website. Klikkt men op de welkompagina op ‘raadpleging’, dan komt men op ‘consult.php’ terecht. Daar wordt eerst d.m.v. specifieke PHP-databankfuncties een lijst samengesteld van alle databanken die zich op de server (de ‘localhost’) bevinden. Hieruit kan men vervolgens een keuze maken. Zo kan men bijvoorbeeld de databank ‘eval0405’ van het schooljaar 2004-2005 gaan raadplegen (opzoeken van oude rapporten). Nadat het systeem geregistreerd heeft dat er een keuze gemaakt is, wordt er nagegaan of de databankstructuur de opvraging van rapporten ondersteunt (voorwaarde is dus dat de databank volgens het nieuwe databankmodel is opgebouwd!). Indien dit niet het geval is, of wanneer het connecteren met de geselecteerde databank niet lukt, dan wordt gevraagd aan de gebruiker om een andere keuze te maken. In het andere geval, bij een succesvolle connectie, wordt een lijst opgevraagd van alle leerlingen in de tabel ‘cursist’. Men kan uit een keuzeveld de eerste letter van de naam van de student selecteren, en daarop volgend worden alle leerlingen waarvan de familienaam met die letter begint overzichtelijk weergegeven (Figuur 4.18), met een knop naast hun naam. Klikken op die knop activeert een hyperlink naar ‘overzichtleerling.php’, waarbij als enige input het stamnummer van de student vereist is. Wanneer de rapportpagina vanuit



Figuur 4.17: Fragment (1e trimester) uit een rapport van de sectie ‘woord’, waar de criteria opgedeeld zijn in categorieën.

het ‘raadpleeg’-gedeelte van de website wordt opgevraagd, dan verschijnt deze in een pop-up venster. Zo kan men snel de rapporten van meerdere leerlingen bekijken. Men kan ook met de terug-knop van de browser terugkeren om een andere letter of databank te selecteren.



Figuur 4.18: In dit voorbeeld werden alle leerlingen waarvan de familienaam begint met de letter ‘D’ opgevraagd uit onze testdatabank.

Hoofdstuk 5

Besluit

Vertrekkende van de probleemstelling werd er een Entity Relationship-diagram opgesteld. Daaruit werd dan het relationeel databankmodel afgeleid. Voor de effectieve implementatie van de database werd rekening gehouden met de gegevens die de tabellen zouden bevolken. Zo werd er gebruik gemaakt van SET om een groep van mogelijke waarden te predefiniëren en rekening gehouden met de grootte van de gegevens. Een integer neemt minder geheugen in beslag dan een float. Zo kon de benodigde ruimte van de databank tot een minimum beperkt worden en werd de snelheid ook hoger. Natuurlijk deze databank is te klein om als gebruiker een verschil te merken in snelheid.

Eenmaal deze zaken er waren, kon het scripting werk beginnen. Een eerste reeks scripts maakt de databank op automatische wijze aan en vult die in vertrekkende van CSV bestanden. Verder is het mogelijk om de databank tot op zekere hoogte, vanuit de browser door middel van de scripts, te beheren. De tweede reeks scripts zijn er om de databank te gebruiken. Docenten kunnen via elektronische weg hun cursisten beoordelen, resultaten van collegadocenten bekijken of van studenten waaraan ze geen les geven.

Toch is dit werk nog vatbaar voor verbetering. Toevoegen van extra functionaliteit waar nog niet aan gedacht is. Het verbeteren van de beveiliging. Zo is een lek in de beveiliging de wijze waarop met sessions gewerkt wordt. Iemand die toegang heeft tot de server van de provider (bijvoorbeeld door er zelf een account te hebben), kan de sessie bestanden lezen en de inhoud ervan veranderen. Dit kan opgelost worden door de sessie objecten te beveiligen. Ook kan het uploaden met bestanden beter. In plaats van ze te kopiëren naar een map is het beter ze in een database te plaatsen. Het voorstel in de thesis om bestanden in een open map te plaatsen die in een enkel uitvoerbare map staat, is niet veel minder veilig dan het plaatsen in een database. Het voorstel is dus afdoende.

Hoofdstuk 6

Bibliografie

- <http://www.w3.org/TR/html401/>
- <http://www.w3schools.com/css/default.asp>
- <http://be2.php.net/>
- <http://www.w3schools.com/php/>
- <http://www.utexas.edu/its/windows/database/datamodeling/index.html>
- <http://dev.mysql.com/doc/refman/4.1/en/>
- <http://www.peachpit.com/articles/article.asp?p=30885&seqNum=7&rl=1>
- <http://latex.ugent.be/>
- Cursus Databanktechnologie, Guy De Tré, Universiteit Gent 2006
- Cursus Internettoepassingen, Peter De Neve, Universiteit Gent 2006
- An introduction to database systems, C.J Date, 7th ed. ,Addison-Wesley,2000