

Programmeerproject Databases 2017-2018

Algemene info

- Projectbegeleiders:
 - o Prof. dr. Bart Goethals – bart.goethals@uantwerpen.be
 - o Dr. Sandy Moens – sandy.moens@uantwerpen.be – G324
 - o M. Sc. Len Feremans – len.feremans@uantwerpen.be – G323
- Groepsproject per 4-5
- Let op: Er is geen tweede zittijd voor dit vak!

Doelstellingen

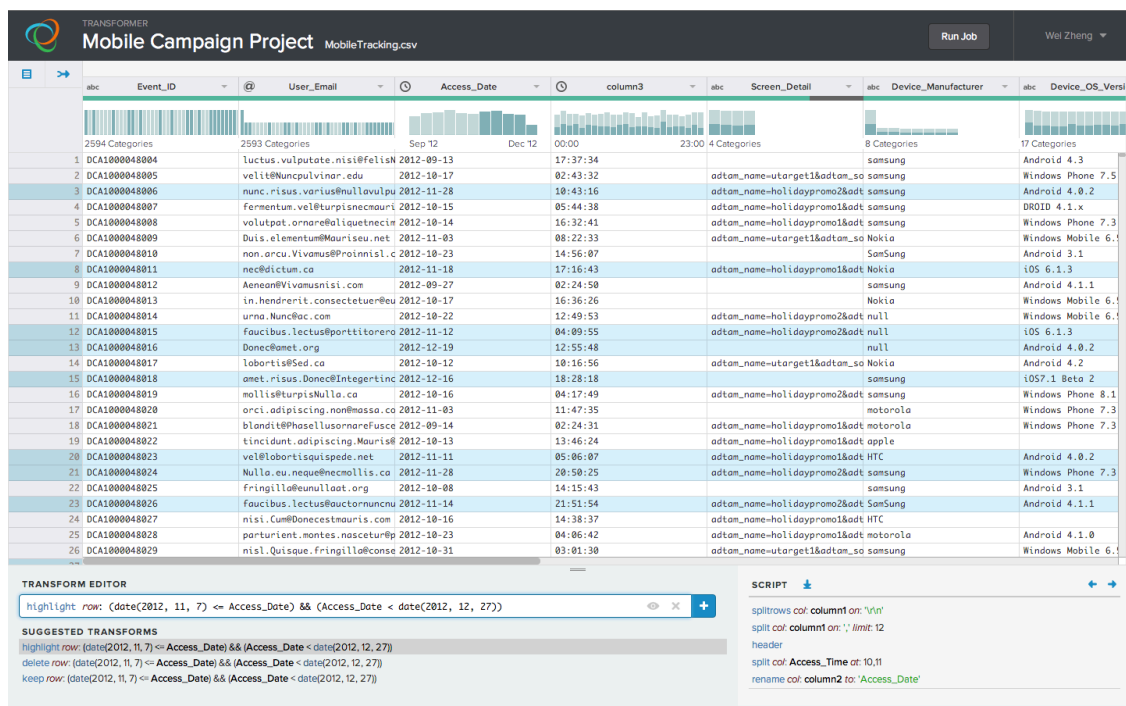
- Een uitgebreid softwareproject tot een goed einde brengen.
- De theorie van het vak “Introduction to Databases” in de praktijk omzetten.
- In groep werken, onafhankelijk van de begeleiding.
- Een nauwkeurige planning maken.
- Duidelijke verslagen schrijven.
- Kwaliteitsvolle bug-vrije software ontwikkelen.
- Creatief denken.

Data cleaner

Data cleaning (ook wel data preprocessing, wrangling of Extract-Load-Transform genoemd) is een van de belangrijke stappen die aan de basis staan van een goede data-analyse. In vele gevallen worden hiervoor verschillende scripts geschreven die een opeenvolging van stappen uitvoeren, zoals bijvoorbeeld het verwijderen van outliers, het opdelen van numerieke data in intervallen, het imputeren van data, etc.

Zulke opkuis van data is vaak saai en repetitief werk dat enkel is weggelegd voor personen met een technische achtergrond. Binnen de onderzoeksgroep willen we een platform bouwen dat het mogelijk maakt om via eenvoudige klikoperaties geavanceerde data cleaning efficiënt uit te voeren op ruwe data.

De uiteindelijke applicatie is een web-based spreadsheet van een (SQL) view op de data, met geavanceerde data cleaning opties. Een voorbeeld van een bestaand commercieel systeem vind je in Figuur 1.



Figuur 1. Screenshot van commerciële data cleaning tool (Trifacta).

Basisvereisten

De applicatie die door jullie zal ontwikkeld worden, bestaat uit een aantal **services** die zo veel mogelijk gebruik dienen te maken van een relationele databank, zowel voor het opslaan van meta informatie als voor ruwe informatie.

De **user-service** wordt gebruikt om nieuwe gebruikers aan te maken en verzoeken te authenticeren:

- Een gebruiker moet een nieuw account kunnen maken op de applicatie op basis van een username en paswoord.
- Een gebruiker moet kunnen inloggen en haar gegevens raadplegen en wijzigen (naam, voornaam, email adres, paswoord).
- De administrator is een speciaal type gebruiker die alle gebruikers kan inspecteren. Het systeem kan meerdere administrators bevatten.
- De administrator is in staat om gebruikers op (non-)actief te zetten alsook om gebruikers te verwijderen.

De **data-import service** beheert het opladen van ruwe data in het systeem:

- Een gebruiker moet ruwe data kunnen opladen vanuit een CSV-bestand [1].
- Een gebruiker kan ook een ZIP-file met meerdere CSV-bestanden opladen.
- Een gebruiker kan ook ruwe data opladen vanuit een bestaande database dump in "standaard" SQL-formaat (Let op: Er bestaan verschillende variaties tussen de verschillende leveranciers van databases).
- Bij het opladen van data bestaande uit verschillende tabellen, moet kunnen worden aangegeven welke tabellen gejoined dienen te worden en op welke kolommen.
- Daarnaast moet een gebruiker metadata kunnen meegeven over de data, bv een naam en een beschrijving.
- Een gebruiker kan enkel haar/zijn eigen data opvragen en informatie over de data wijzigen.
- Data kan toebehoren aan meerdere gebruikers met verschillende rechten.

De **data-transform service** wordt gebruikt om bestaande data op te kuisen, met elkaar te koppelen en te transformeren:

- De gebruiker moet data kunnen selecteren voor opkuis (zowel ruw als reeds verwerkte data).
- De gebruiker moet per attribuut kunnen:
 - o aangeven wat het type van het attribuut is;

- o aangeven welke operatie dient uitgevoerd te worden op het attribuut. Hierbij zijn er afhankelijk van het type van get attribuut verschillende mogelijkheden. In Tabel 1 tonen we een overzicht van acties die jullie moeten ondersteunen);
 - o sorteren op attribuut (ascending/descending) (in view).
- De gebruiker moet rijen kunnen verwijderen op basis van een eenvoudig predicaat dat gebruik maakt van de meest voorkomende logische operatoren (=, >, <, CONTAINS, AND, OR, NOT, etc.);
- Imputeren van ontbrekende data (invullen van lege waarden) obv:
 - o Gemiddelde waarde
 - o Mediaan
- Houd een historiek bij van alle uitgevoerde transformaties.

Type attribuut	Ondersteunde acties
Elk type	<ul style="list-style-type: none"> - Verwijderen attribuut - Find-and-replace - Type veranderen (e.g. string naar int, string naar datum/tijd)
Categorisch	<ul style="list-style-type: none"> - One-hot-encoding [5]
Numeriek	<ul style="list-style-type: none"> - Normaliseren: herschalen tussen 0 en 1 aan de hand van z-waarden [6] - Discretiseren: <ul style="list-style-type: none"> o equi-distant intervallen, zoals in een histogram [7,8] o equi-frequent intervallen [8] o handmatige ranges - Outliers verwijderen: waarden kleiner of groter dan voorgestelde waarde verwijderen
Datum/Tijd	<ul style="list-style-type: none"> - Parsen van datum/tijd - Extraheren van delen van de datum, bijvoorbeeld dag van de week, maand van het jaar, jaar, etc.
Tekst	<ul style="list-style-type: none"> - Find-and-replace a.h.v. reguliere expressies

Tabel 1: Overzicht van verschillende transformaties op attribuut niveau

De **view-service** biedt de mogelijkheid om zowel ruwe data als verwerkte data te bekijken:

- De basis view biedt een tabel lay-out van de data.
- Per kolom geeft een grafiek meer informatie over de hele data, bv een histogram voor een numeriek of categoriek attribuut.
- Per kolom kunnen er ook verschillende statistieken berekend worden, zoals de meest-voorkomende waarde, minimum/maximum/gemiddelde van numerieke kolommen, aantal waarden die leeg zijn, etc.
- De view biedt de mogelijkheid om de data te downloaden naar CSV, met keuze van scheidingsteken, quote karakter, karakter voor lege waardes, etc.

Verder moet het systeem ook **efficiënt** kunnen omgaan met grote bestanden:

- Indien een actie eenvoudig te implementeren is in SQL, zoals een projectie op kolommen of een selectie van rijen op basis van een filter, moeten jullie achterliggend een query (of SQL VIEW) gebruiken.

- Complexere acties zoals normalisatie of discretisatie, kunnen in memory gebeuren met behulp van daarvoor gespecialiseerde bibliotheken [2,3], maar bewaar het resultaat steeds in de databank.

Opgelet!

- Gebruikers in het systeem dienen uniek te zijn.
- Het systeem moet zelf opkuisend zijn, in die zin dat het verwijderen van bijvoorbeeld een gebruiker, alle bijhorende data verwijdert.
- De ruwe data moet altijd behouden blijven.
- Let erop dat communicatie en input voor het hele project veilig gebeuren. Dat wil zeggen dat het systeem behoed is tegen bijvoorbeeld SQL-injectie aanvallen [4].

Extra functionaliteit

De basisvereisten tellen mee voor ongeveer de helft van de punten en worden toegekend na de 2^{de} tussentijdse presentatie (zie verder Evaluatie). De resterende punten krijgen jullie op basis van aanpassingen en extra functionaliteit na deze presentatie. Het is hierbij belangrijk dat jullie zo goed mogelijk de verwachtingen en feedback weten om te zetten in een goed-werkende applicatie.

Enkele mogelijke voorbeelden van extra functionaliteit zijn meer geavanceerde transformaties, visualisatie technieken, optimalisaties en een rijkere database, we denken hierbij onder andere aan:

- Omzetting van tekst naar features, bijvoorbeeld met bag-of-words [9], opdat je tekst in dit formaat ook kan gebruiken voor voorspellingen te doen, gebruik makende van machine learning [3].
- Data-deduplicatie, waarmee je met een slimme similarity maat, zoals edit-distance, zoekt naar tekst-waarden die mogelijk hetzelfde betekenen (e.g. "Antwerpen" en "Aentwaerpen") [10].
- Imputeren van ontbrekende waarden, of incorrecte waarde, op basis van een voorspelde waarde [3]. Een ontbrekende waarde voorspellen kan door gebruik te maken van machine learning algoritmen, zoals k-nearest neighbors.
- Aggregaat transformaties, berekenen met bijvoorbeeld GROUP BY het aantal, de som, het gemiddelde van kolom B t.o.v. kolom A.
- Zorg ervoor dat de weergave van grote datasets *gepagineerd* gebeurt, dus laadt enkel data die effectief weergegeven wordt.
- Implementeer de transformaties *streaming*.
- Advanced handmatige discretisatie.

Uitvoering van het Project

Groep samenstelling

Dit project is bedoeld voor groepen van 4 (maximaal 5) studenten. De groepen worden door jullie *zelf samengesteld* en aan ons gemeld (mailen naar len.feremans@uantwerpen.be). Studenten die na de deadline nog steeds geen groep hebben zullen door ons in groepen ingedeeld worden. Hierbij bestaat de mogelijkheid dat de bestaande groepen een extra lid krijgen. De officiële groepslijsten worden via Blackboard bekend gemaakt.

Versie controle

Alle documentatie, SQL-code en programmatie code moeten jullie bijhouden in een online *versie controle systeem* (bijvoorbeeld GitHub). Jullie maken een account, en geven ons toegang tijdens de eerste weken. Wat de organisatie van jullie bestanden betreft, verwachten we een heldere structuur, met bron code, SQL-code en documentatie in afzonderlijke mappen. Daarnaast verwachten we een readme met een woordje uitleg.

Wekelijkse status update

We verwachten dat jullie elke lesweek een *vergadering* houden waarbij de status en de planning van jullie project wordt besproken. Het resultaat hiervan is een *wekelijks verslag* met aanwezigheden, opvolging geplande taken, etc. volgens de *template* op Blackboard. De elementen aanwezig in deze template moeten minimaal in jullie verslagen aanwezig zijn. We zullen deze verslagen gebruiken om eventuele *problemen* binnen de groepen tijdig op te merken, en eventueel ook als motivatie voor het geven van individuele punten.

Een lokaal is gereserveerd *woensdag van 08.30 tot 15.00*. Tussen *10.00 en 11.00* is een assistent aanwezig en kunnen vragen en problemen bij de uitvoering van jullie project besproken worden.

Rapporten

Daarnaast verwachten we dat 3 keer een *rapport* wordt ingediend (zie ook agenda verderop in dit document voor de exacte deadlines). Voor deze rapporten stellen we jullie ook een *template* ter beschikking via Blackboard. De elementen aanwezig in deze template moeten minimaal in jullie rapporten aanwezig zijn, waaronder het *design* van het programma als geheel, een ER-diagram van de databank en een beschrijving van de functionaliteit, alsook een overzicht van de afgewerkte taken van elk teamlid. Let op dat elk teamlid bij de implementatie betrokken moet zijn, en niet enkel bij het project beheer of het maken van documentatie.

Presentaties

Voor de *presentaties* verwachten we een *werkende demonstratie*, waarbij jullie feedback krijgen van de jury. Een groot deel van de punten zal

gebaseerd zijn op het al dan niet werken van de vereiste functionaliteit. Tijdens het semester zullen er *2 tussentijdse presentaties* georganiseerd worden, gevolgd door een *eindpresentatie* tijdens de examenperiode van een *online beschikbare webapplicatie*. Hiervoor mag je gebruik maken van je eigen hosting, of kan je contact opnemen met Muriel (muriel.dejonghe@uantwerpen.be) om je project te hosten op de studenten server. Een demo mag je op je eigen laptop doen, maar we raden sterk aan dat jullie op voorhand alles grondig controleren (internetverbinding, connectie met projector, etc.) opdat de demo *vlekkeloos* verloopt.

Voor de *eerste tussentijds presentatie* verwachten we dat alle keuzes gemaakt zijn rond het ontwerp, de database en de taakverdeling. Voorzie *mockups* (pagina's zonder achterliggende koppeling met de databank of a.h.v. tekeningen) voor pagina's die nog niet geïmplementeerd zijn.

Voor de tweede tussentijdse presentatie worden jullie *finaal geëvalueerd* op basis van een online beschikbare demo van de applicatie, *dat voldoet aan al de basisvereisten*.

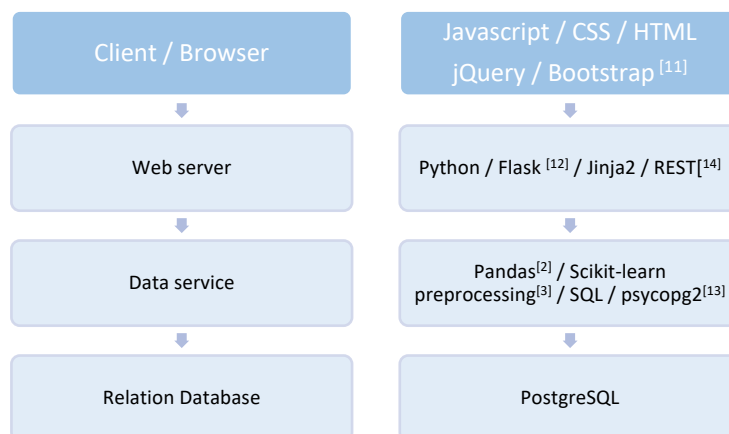
Eind mei verwachten we van jullie een *finale versie* indienen en dat een *online versie* beschikbaar is.

Opgelet: zorg dat iedereen aan bod komt bij de presentaties!

Technologische stack

Er zijn verschillende technologieën nodig om een web applicatie te maken. We vragen dat jullie gebruik maken van de libraries, frameworks en tools beschreven in Figuur 2 [2,3,11,12,13,14]. Daarnaast voorzien we een tutorial met voorbeeld code [15], losjes gebaseerd op deze Flask tutorial [12].

Let op: Indien jullie als groep een andere library, programmeertaal of databank willen gebruiken, moeten jullie dit tijdens het wekelijks contactmoment motiveren.



Figuur 2: Technologie stack.

Evaluatie

Naast de functionaliteit worden jullie ook beoordeeld op de volgende criteria:

- teamwork
- kwaliteit van de wekelijkse verslaggeving, rapporten en presentaties
- kwaliteit van de programma-code
- kwaliteit van het databank ontwerp en de SQL queries
- documentatie en unit testen in de bron code
- kwaliteit van de online demonstraties
- gebruiksvriendelijkheid van de applicatie.

Bij een eerlijke taakverdeling, en als iedereen in staat is zijn deel te presenteren en vragen te beantwoorden, krijgen alle leden van de groep dezelfde punten.

Het eindresultaat wordt berekend aan de hand van de volgende score tabel:

Punten	
Basisvereisten (April)	
7	Correctie en volledige implementatie basisvereisten met online demonstratie. Voorbeelden minpunten: login/multi-user functionaliteit werkt niet; ontbreken van transformaties uit tabel 1; visualisatie bestaat enkel uit basistabel zonder histogrammen, ...
1	Gebruiksvriendelijkheid/user interface. Voorbeelden minpunten: interactie aan de hand van lange formulieren op afzonderlijke pagina's i.p.v. integratie binnen 1 pagina; onduidelijk hoe basis functionaliteit werkt.
1	Code/database kwaliteit. Voorbeelden minpunten: studenten hebben niet samengewerkt aan 1 consistente code basis; 1, 2 of 3 grote klassen met alle code in, in de plaats van decompositie in verschillende klassen of modules; geen duidelijk onderscheid in model/view/controller logica; databank schema niet gemaakt aan de hand van correct E.R. diagram; SQL queries of schema niet correct of onnodig complex.
Extra functionaliteit (Mei)	
5	Interessante, creatieve en goed werkende extra uitbreidingen. Voorbeelden minpunten: enkel basisvereisten afgewerkt; extra functionaliteit te eenvoudig.
1	Gebruiksvriendelijkheid/user interface.
2	Code/database kwaliteit.
Project management (Semester)	
1	Indiening wekelijkse rapporten volgens template. Voorbeelden minpunten: studenten die rapporten maken aan het einde van maand 1 of 2; in versie controle systeem enkel updates tegen deadline van het project.
1	Rapporten. Voorbeelden minpunten: heel summier rapport, of template niet volledig ingevuld; geen UML schema; E.R. diagram komt niet overeen met de applicatie.
1	Demonstratie. Voorbeelden minpunten: niet iedereen is aan bod gekomen, presentatie bestaat enkel uit bullets en is afgelezen of onduidelijk; geen bespreking van interessante algoritmes of uitdagende queries.
/ 20	

Agenda

Taak	Deadline
Toelichting project en voorbeeld ontwerp webapplicatie	14/02 8:30
Voorstel groepsindeling (via e-mail)	15/02 12:00
Wekelijks verslag (via versie controle systeem)	Vanaf week 2, elke lesweek
Wekelijkse begeleiding	Elke lesweek, woensdag, 10-11
Indienen 1 ^{ste} tussentijds rapport	13/03 23:59
1 ^{ste} tussentijdse presentatie	14/03
Indienen 2 ^{de} tussentijds rapport	24/04 23:59
2 ^{de} tussentijdse presentatie + online demo beschikbaar dat voldoet aan basis vereisten	25/04
Indienen finaal rapport + online demo beschikbaar met extra functionaliteit	23/05 23:59
Finale presentatie	28/05

Referenties

- [1] CSV formaat:
https://nl.wikipedia.org/wiki/Kommagescheiden_bestand
- [2] Pandas (Python data analysis framework), 10 minute tutorial:
<https://pandas.pydata.org/pandas-docs/stable/10min.html>
- [3] Sci-kit learn preprocessing library:
<http://scikit-learn.org/stable/modules/preprocessing.html#preprocessing>
- [4] SQL Injectie uitleg:
https://www.w3schools.com/sql/sql_injection.asp
- [5] One-hot-encoding blog met uitleg:
<https://machinelearningmastery.com/how-to-one-hot-encode-sequence-data-in-python/>
- [6] Z-score uitleg:
<https://nl.wikipedia.org/wiki/Z-score>
- [7] Histogram uitleg:
<https://en.wikipedia.org/wiki/Histogram>
- [8] Equal width/frequency discretisatie uitleg:
http://www.saedsayad.com/unsupervised_binning.htm
- [9] Uitleg bag-of-words voor representatie tekst:
https://en.wikipedia.org/wiki/Bag-of-words_model
- [10] Link naar python dedupe package for deduplicatie:
<https://dedupe.io/developers/library/en/latest/How-it-works.html>
- [11] Tutorial HTML, CSS, Bootstrap, Javascript en jQuery:
<https://www.w3schools.com/>
- [12] Tutorial Flask library om webapplicatie mee te maken:
<https://pythonspot.com/flask-web-app-with-python/>
- [13] Uitleg psycopg2 library om SQL queries uit te voeren:
https://wiki.postgresql.org/wiki/Using_psycopg2_with_PostgreSQL
- [14] Uitleg REST:
<https://www.ibm.com/developerworks/library/ws-restful/index.html>
- [15] Tutorial Programmeer Project:
<https://github.com/lfereman/tutorial>