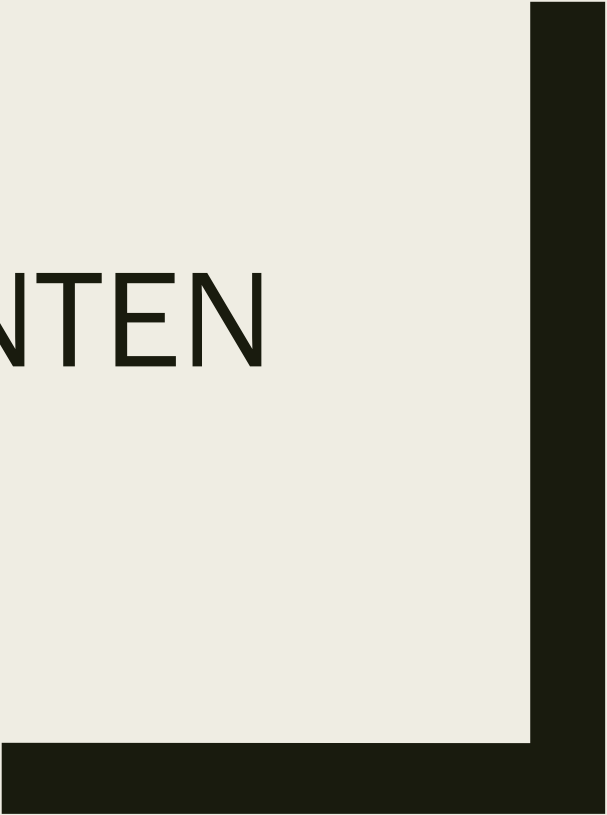




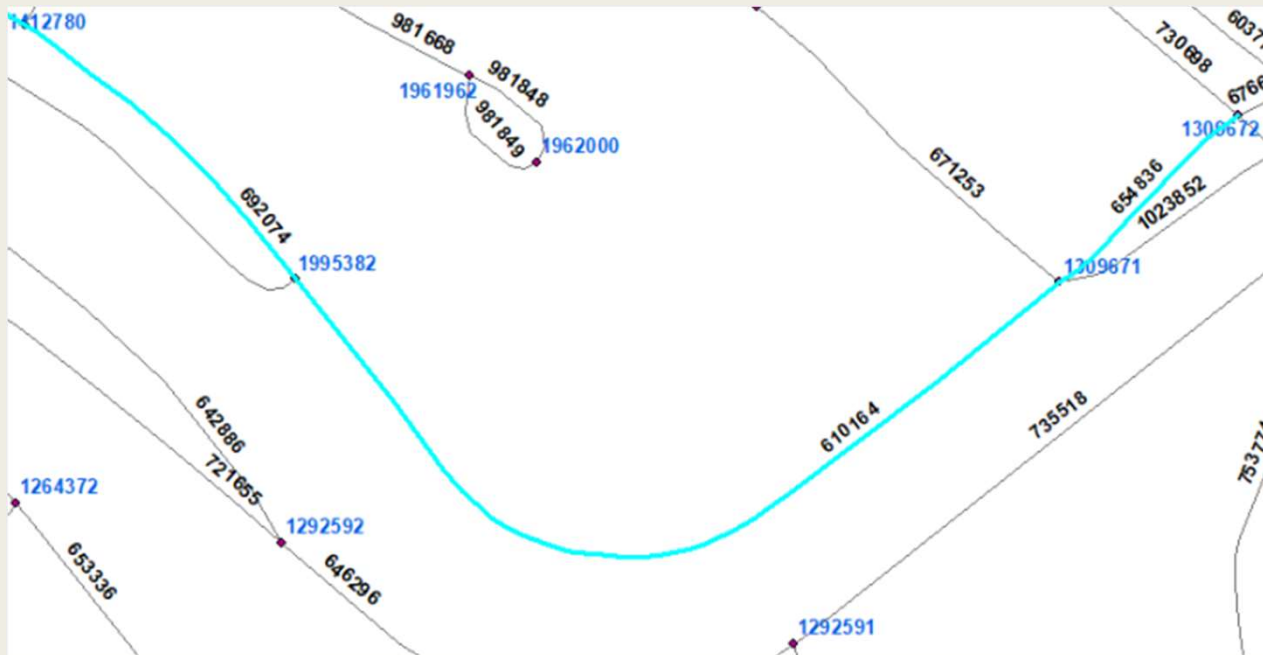
STRATENSEGMENTEN

Programmeren – Labo
Matthijs De Riek



Structuur

- Probleemstelling
- Datamodel
- Tool 1: Inlezen
- Tool 2: Database creëren + data wegschrijven
- Tool 3: Bevragen van de database



Probleemstelling

Gegeven

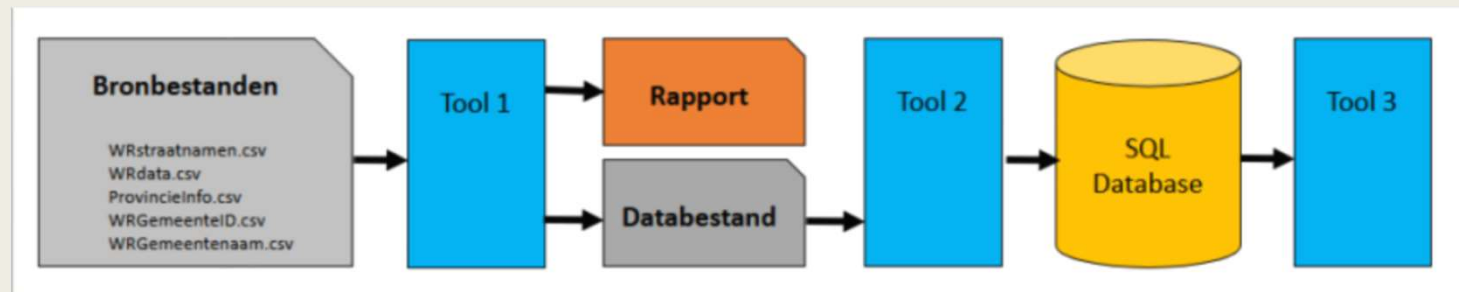
- Bestanden (.csv) van de Belgische overheid met alle posities van alle straten in België en hun namen.

1. WRdata.csv
2. WRGemeentelD.csv
3. WRStraatnamen.csv
4. WRGemeentenaam.csv
5. ProvincieInfo.csv

Probleemstelling

Gevraagd

- Tool die bronbestanden inleest (1)
- Tool die bestanden wegschrijft naar database (2)
- Tool die deze database kan bevragen (3)



Datamodel

SegmentId, Punten, Knopen, StraatId



WRData.csv

```
wegsegmentID;geo;morfologie;status;beginWegknoopID;eindWegknoopID;linksStraatnaamID;rechtsStraatnaamID  
1;LINESTRING (217368.75 181577.0159999989, 217400.1099999994 181499.5159999989);114;4;126722;41353;504;1202
```

StraatId, Straatnaam



WRStraatnamen.csv

```
EXN;LOS  
1;Acacialaan
```

GemeentId, GemeenteNaam, Taalcode



WRGemeentenaam.csv

```
gemeenteNaamId;gemeenteId;taalCodeGemeenteNaam;gemeenteNaam  
1;1;nl;Aartselaar
```

StraatId, GemeentId



WRGemeentId.csv

```
straatNaamId;gemeenteId  
1;1
```

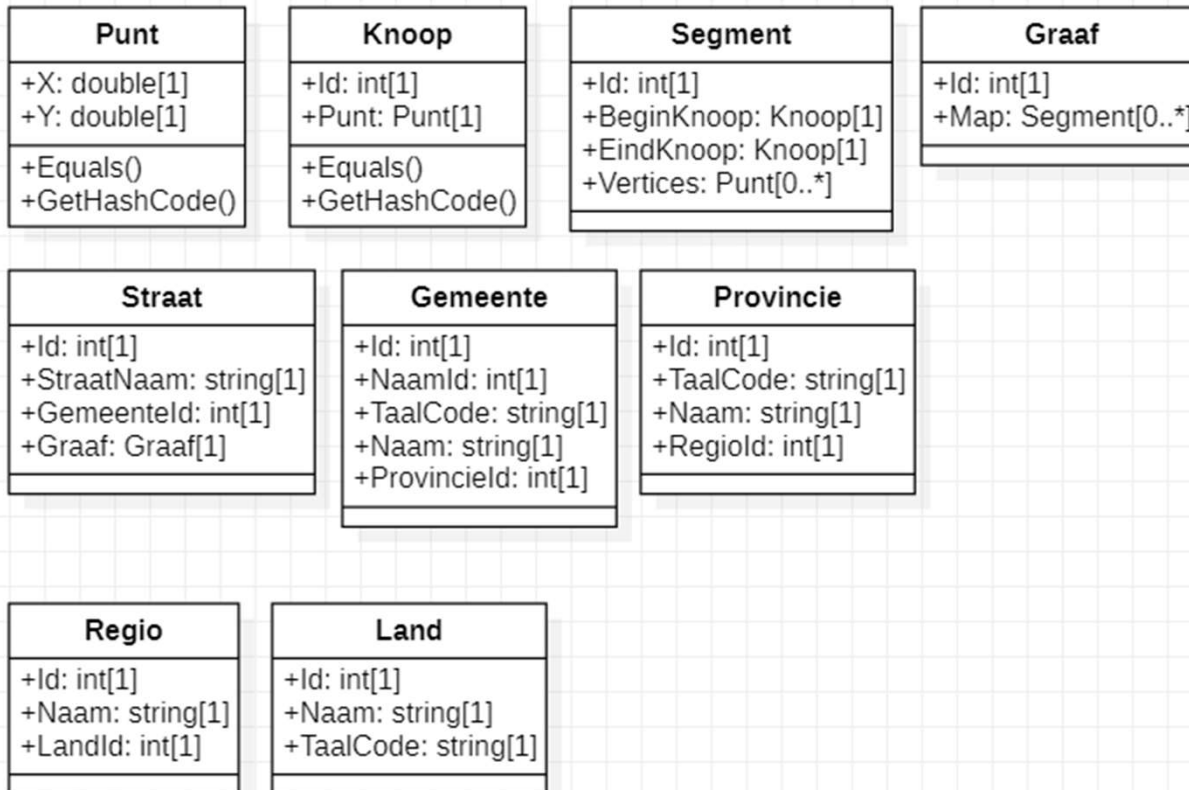
GemeentId, ProvincieId, Taalcode,
ProvincieNaam



ProvincieInfo.csv

```
gemeenteId;provincieId;taalCodeProvincieNaam;provincieNaam  
1;1;nl;Antwerpen
```

Datamodel



- **Object Oriented**

Data wordt geordend via het creëren van objecten

- **Database gericht model**

Maakt gebruik van ID i.p.v. referenties naar effectieve objecten met oog op implementatie van de databank

- **Extra klassen**

Maken het model uitbreidbaarder

Tool 1 : Inlezen van bestanden

Providers

| WRDataProvider |
|---|
| - _straatIdSegmenten: Dictionary <int List<Segment>>[1] |
| -Read() +GetSegmentListByStraatId(int straatId): List<Segment> |

| GemeenteProvincieProvider |
|---|
| - _gemeentes: Dictionary<int Gemeente>[1] - _provincies: Dictionary<int Provincie>[1] - _proldGemeenten: Dictionary<int List<Gemeente>> - _taalCode: string[1] |
| -Read() -GetGemeenteById(int gemId): Gemeente +GetProvincieWithoutRegioId(int prold): Provincie +GetGemeentenByProvincieId(int prold): List<Gemeente> |

| StratenProvider |
|--|
| - _wrDataProvider: WRDataProvider[1] - _idStraatNaam: Dictionary<int string>[1] - _gemIdStraten: Dictionary<int List<Straat>>[1] |
| +GetStratenByGemId(int gemeenteld): List<Straat> -Read() |

| RegioProvider |
|---|
| - _regioIdListProvincies: Dictionary <int List<Provincie>>[1] - _taalCode: string[1] |
| -Read() +GetProvinciesByRegioId(int id): List<Provincie> |

Tool 1 : Inlezen van bestanden

Rapport

| Rapport |
|---|
| -_regioProvider: RegioProvider[1] -_gemeenteProvincieProvider: GemeenteProvincieProvider[1] -_stratenProvider: StratenProvider[1] |
| +Rapporteur(string path) -GetSpaces(int n): string[1] |

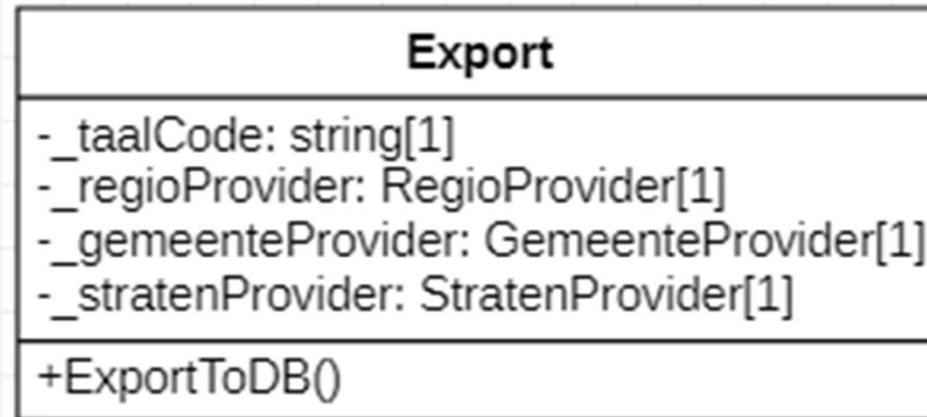
Tool 2: Wegschrijven naar database

Exporters

| Exporters |
|--|
| <ul style="list-style-type: none">-_connString: string[1]-_wegGeschrevenKnoopIdPuntId: Dictionary<int int>[1]-_weggeschrevenSegmenten: HashSet<int>[1] |
| <ul style="list-style-type: none">-ExportPunt(Punt p): int[1]-ExportKnoop(int knoopId, int puntId): int[1]-ExportSegment(Segment s): int[1]-ExportGraaf(Graaf g)-ExportTussenTabel(int id1, int id2, string tabelNaam)+ExportStraat(Straat s)+ExportGemeente(Gemeente g)+ExportProvincie(Provincie p)+ExportRegio(Regio r)+ExportLand(Land l) |

Tool 2: Wegschrijven naar database

Export



Tool 2: Wegschrijven naar database

Database: basis

```
CREATE TABLE Punt(  
  Id int IDENTITY (1,1) NOT NULL,  
  X decimal(15,9),  
  Y decimal(17,11),  
  PRIMARY KEY (Id)  
);  
  
CREATE TABLE Knoop(  
  Id int NOT NULL,  
  PuntId int,  
  PRIMARY KEY (Id)  
);  
  
CREATE TABLE Segment(  
  Id int not null,  
  BeginKnoopId int,  
  EindKnoopId int,  
  PRIMARY KEY (Id),  
);  
  
CREATE TABLE Graaf(  
  Id int not null,  
  PRIMARY KEY (Id)  
);
```

```
CREATE TABLE Straat(  
  Id int NOT NULL,  
  StraatNaam NVARCHAR(500),  
  GraafId int,  
  GemeenteId int,  
  PRIMARY KEY (Id)  
);  
  
CREATE TABLE Gemeente(  
  Id int not null,  
  NaamId int,  
  TaalCode NVARCHAR(50),  
  Naam NVARCHAR(500),  
  ProvincieId int,  
  PRIMARY KEY (Id)  
);  
  
CREATE TABLE Provincie(  
  Id int not null,  
  TaalCode NVARCHAR(50),  
  Naam NVARCHAR(500),  
  RegioId int,  
  PRIMARY KEY (Id)  
);
```

```
CREATE TABLE Land(  
  Id int not null,  
  Naam NVARCHAR (500),  
  TaalCode NVARCHAR (50),  
  PRIMARY KEY (Id)  
);  
  
CREATE TABLE Regio(  
  Id int not null,  
  Naam NVARCHAR(500),  
  LandId int,  
  PRIMARY KEY (Id)  
);
```

Database : Tussentabellen

```
CREATE TABLE GraafId_SegmentId(  
  GraafId int not null,  
  SegmentId int not null  
);  
ALTER TABLE GraafId_SegmentId  
  ADD PRIMARY KEY (GraafId, SegmentId);  
  
CREATE TABLE SegmentId_PuntId(  
  SegmentId int not null,  
  PuntId int not null,  
);  
ALTER TABLE SegmentId_PuntId  
  ADD PRIMARY KEY (SegmentId, PuntId);
```

Tool 2:
Wegschrijven naar
database

Database : FK-Constraints

Tool 2: Wegschrijven naar database

```
ALTER TABLE Straat
ADD CONSTRAINT fk_gemId_straat FOREIGN KEY (GemeenteId) REFERENCES Gemeente(Id);
ALTER TABLE Gemeente
ADD CONSTRAINT fk_proId_gem FOREIGN KEY (ProvincieId) REFERENCES Provincie(Id);
ALTER TABLE Provincie
ADD CONSTRAINT fk_reigId_pro FOREIGN KEY (RegioId) REFERENCES Regio(Id);
ALTER TABLE Regio
ADD CONSTRAINT fk_landId_reg FOREIGN KEY (LandId) REFERENCES Land(Id);
ALTER TABLE Knoop
ADD CONSTRAINT fk_puntId_Knoop FOREIGN KEY(PuntId) references Punt (Id);
ALTER TABLE Segment
ADD CONSTRAINT fk_beginknoopId_Segment FOREIGN KEY(BeginKnoopId) references Knoop (Id);
ALTER TABLE Segment
ADD CONSTRAINT fk_eindknoopId_Segment FOREIGN KEY(EindKnoopId) references Knoop (Id);
ALTER TABLE SegmentId_PuntId
ADD CONSTRAINT fk_puntID_segmentId_PuntId FOREIGN KEY (PuntId) REFERENCES Punt (Id);
ALTER TABLE Straat
ADD CONSTRAINT fk_graafId_Straat FOREIGN KEY(GraafId) REFERENCES Graaf(Id);
```

Tool 3: Bevragen van de database

Importers

| Importers |
|---|
| -_connString: string[1] |
| +GetStraatByStraatnaamEnGemeentenaam(string straatNaam, string gemeenteNaam): Straat[1] |
| +GetStraatById(int straatId): Straat[1] |
| -GetAllPointsOfSegment(Segment s, SqlConnection openConnection): List<Punt>[1] |
| +GetListStraatIdByGemeenteNaam(string gemeenteNaam): List<int>[1] |
| +GetListStraatNamenByGemeenteNaam(string gemeenteNaam): List<string>[1] |
| +GetGemeenteNaamEnProvincieNaamByGemeenteld(int gemeenteld): Tuple<string string |
| +GetGemeentesIdNaamByProvincieNaam(string provincieNaam): List<Gemeente>[1] |
| +GetAdjacentStraten(Straat s): List<Straat>[1] |
| -GetAdjacentStraatIdsInGemeente(Straat s): HashSet<int>[1] |
| +GetAllKnoopIdsFromStraatId(int straatId): List<int>[1] |
| -GetPuntByKnoopId(int knoopId, SqlConnection openConnection): Punt[1] |
| -SortVertices(Segment s): List<Punt>[1] |
| -SortMap(List Segment segmenten): List<Segment>[1] |
| -ReverseSegment(Segment s): Segment[1] |

Print

Tool 3: Bevragen
van de database

Print

- +PrintKruisendeStraten(int straatId)
- +PrintLijstStraatNamen(string gemeenteNaam)
- +PrintLijstStraatIds(string gemeenteNaam)
- +PrintStraat(string straatNaam, string gemeenteNaam)
- +PrintStraat(int straatId)
- +PrintProvincieInfo(string provincieNaam)
- PrintStraat(Straat s)
- GetHeader(string message)
- GetFooter(string message)
- GetXchars(int n, char c): string[1]