Opdracht 5 - ORM

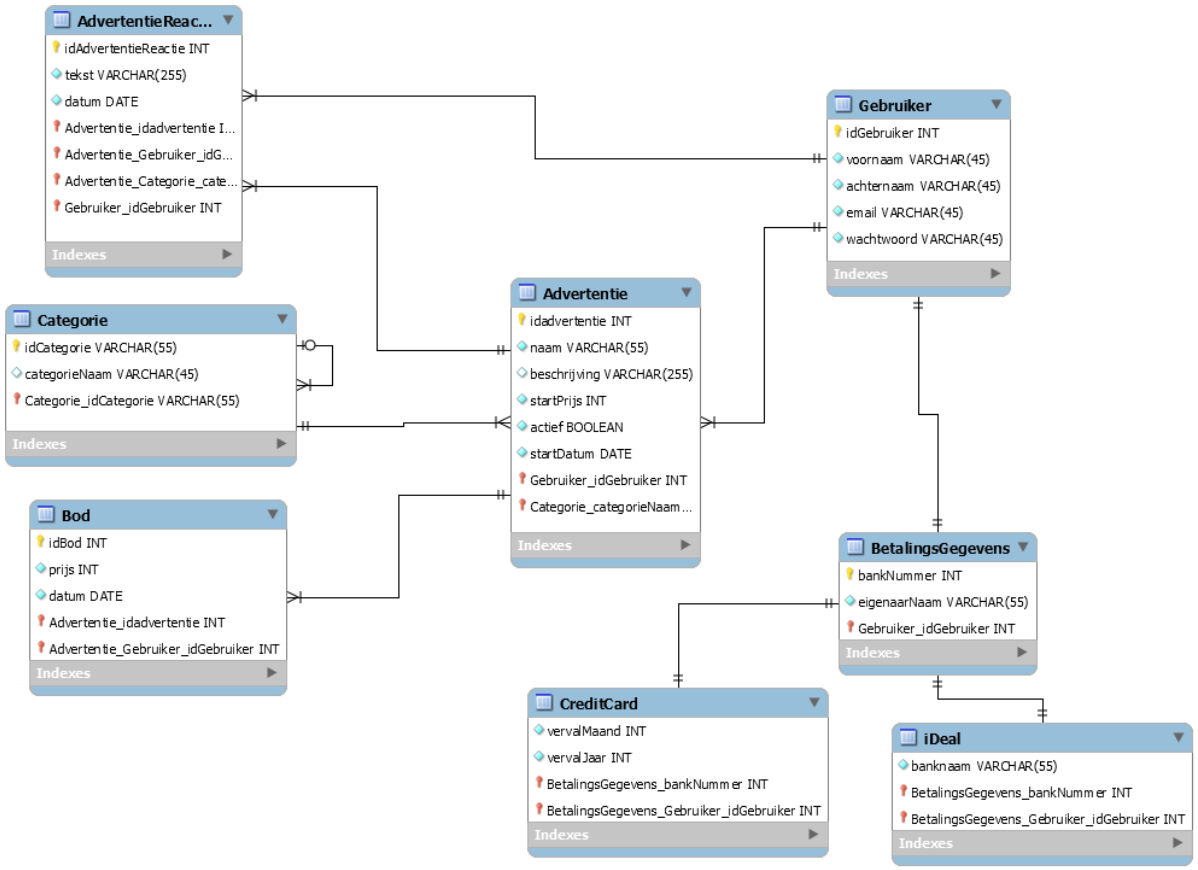
Verslag

Analyse 8 - Advanced Databases (2014-2015)

Namen:	HoYe Lam, Rinesh Ramadhin
Studentnummer:	0876814, 0882447
Klas:	INF2D
Vak:	INFANL01-8
Opdracht:	ORM
Datum:	24 – 06 - 2015

Inhoud

Database structuur



We hebben het klassendiagram, zoals aangegeven in het opdracht document onder het kopje “Case”, omgezet in een ERD. Ook hebben we, waar nodig, kleine aanpassingen gemaakt voor de opdrachten en scenario’s. Ook hebben we de keys in creditCard en iDeal omgewisseld. Dit omdat het zo logischer is.

Java classes

In iedere klas mappen we de tabel aan de postgreSQL database. We geven aan wat de colums zijn, en als er relaties zijn met andere tabellen geven we ook aan wat voor relatie en welke foreign keys er zijn. We gebruik annotations om dit te bereiken. Hieronder vindt u de code van elke class.

Advertentie

```java
package entities;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.ManyToOne;
import javax.persistence.Table;

@Entity
@Table(name = "Advertentie")
public class Advertentie {

    private Integer idAdvertentie;
    private String naam;
    private String beschrijving;
    private Integer startPrijs;
    private boolean actief;
    private String startDatum;
    private Gebruiker gebruiker;
    private Categorie categorie;

    public Advertentie() {
    }

    public Advertentie(String naam, String beschrijving, Integer startPrijs,
            boolean actief, String startDatum, Gebruiker gebruiker, Categorie categorie) {
        this();
        this.naam = naam;
        this.beschrijving = beschrijving;
        this.startPrijs = startPrijs;
        this.actief = actief;
        this.startDatum = startDatum;
        this.gebruiker = gebruiker;
        this.categorie = categorie;
    }
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name="idAdvertentie")
    public Integer getIdAdvertentie() {
        return idAdvertentie;
    }

    public void setIdAdvertentie(Integer idAdvertentie) {
        this.idAdvertentie = idAdvertentie;
    }

    @Column(name="naam")
    public String getNaam() {
        return naam;
    }

    public void setNaam(String naam) {
        this.naam = naam;
    }

    @Column(name="beschrijving")
    public String getBeschrijving() {
        return beschrijving;
    }

    public void setBeschrijving(String beschrijving) {
        this.beschrijving = beschrijving;
    }
```

```
    @Column(name="startPrijs")
    public Integer getStartPrijs() {
        return startPrijs;
    }

    public void setStartPrijs(Integer startPrijs) {
        this.startPrijs = startPrijs;
    }

    @Column(name="actief")
    public boolean isActief() {
        return actief;
    }

    public void setActief(boolean actief) {
        this.actief = actief;
    }

    @Column(name="startDatum")
    public String getStartDatum() {
        return startDatum;
    }

    public void setStartDatum(String startDatum) {
        this.startDatum = startDatum;
    }

    @ManyToOne(cascade = CascadeType.ALL)
    public Gebruiker getGebruiker() {
        return gebruiker;
    }

    public void setGebruiker(Gebruiker gebruiker) {
        this.gebruiker = gebruiker;
    }

    @ManyToOne(cascade = CascadeType.ALL)
    public Categorie getCategorie() {
        return categorie;
    }

    public void setCategorie(Categorie categorie) {
        this.categorie = categorie;
    }
}
```

AdvertentieReactie

```
package entities;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.ManyToOne;
import javax.persistence.Table;

@Entity
@Table(name="AdvertentieReactie")
public class AdvertentieReactie {
    private Integer idAdvertentieReactie;
    private String tekst;
    private String datum;
    private Advertentie advertentie;
    private Gebruiker gebruiker;


    public AdvertentieReactie() {
    }

    public AdvertentieReactie(String tekst, String datum, Advertentie advertentie, Gebruiker gebruiker) {
        this();
        this.tekst = tekst;
        this.datum = datum;
        this.advertentie = advertentie;
        this.gebruiker = gebruiker;
```

```
    }

    @Id
    @GeneratedValue
    @Column(name="idAdvertentie")
    public Integer getIdAdvertentieReactie() {
        return idAdvertentieReactie;
    }

    public void setIdAdvertentieReactie(Integer idAdvertentieReactie) {
        this.idAdvertentieReactie = idAdvertentieReactie;
    }

    @Column(name="tekst")
    public String getTekst() {
        return tekst;
    }

    public void setTekst(String tekst) {
        this.tekst = tekst;
    }

    @Column(name="datum")
    public String getDatum() {
        return datum;
    }

    public void setDatum(String datum) {
        this.datum = datum;
    }

    @ManyToOne(cascade = CascadeType.ALL)
    public Advertentie getAdvertentie() {
        return advertentie;
    }

    public void setAdvertentie(Advertentie advertentie) {
        this.advertentie = advertentie;
    }

    @ManyToOne(cascade = CascadeType.ALL)
    public Gebruiker getGebruiker() {
        return gebruiker;
    }

    public void setGebruiker(Gebruiker gebruiker) {
        this.gebruiker = gebruiker;
    }

}
```

BetalingsGegevens

```
package entities;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToOne;
import javax.persistence.PrimaryKeyJoinColumn;
import javax.persistence.Table;
import entities.Gebruiker;
import org.hibernate.annotations.GenericGenerator;
import org.hibernate.annotations.Parameter;

@Entity
@Table(name = "BetalingsGegevens")
public class BetalingsGegevens {
    private Integer idGebruiker;
    private Integer bankNummer;
    private String eigenaarNaam;
    private Gebruiker gebruiker;

    public BetalingsGegevens() {
```

```
    }

    public BetalingsGegevens(Integer bankNummer, String eigenaarNaam,
            Gebruiker gebruiker) {
        this();
        this.bankNummer = bankNummer;
        this.eigenaarNaam = eigenaarNaam;
        this.gebruiker = gebruiker;
    }

    @Id
    @GeneratedValue(generator = "gebruiker")
    @GenericGenerator(name = "gebruiker", strategy = "foreign", parameters = @Parameter(value = "gebruiker", name =
"property"))
    public Integer getIdGebruiker() {
        return idGebruiker;
    }

    public void setIdGebruiker(Integer idGebruiker) {
        this.idGebruiker = idGebruiker;
    }

    @Column(name = "bankNummer")
    public Integer getBankNummer() {
        return bankNummer;
    }

    public void setBankNummer(Integer bankNummer) {
        this.bankNummer = bankNummer;
    }

    @Column(name = "eigenaarNaam")
    public String getEigenaarNaam() {
        return eigenaarNaam;
    }

    public void setEigenaarNaam(String eigenaarNaam) {
        this.eigenaarNaam = eigenaarNaam;
    }

    @OneToOne(cascade = CascadeType.ALL, fetch= FetchType.LAZY)
    @JoinColumn(name="idGebruiker")
    public Gebruiker getGebruiker() {
        return gebruiker;
    }

    public void setGebruiker(Gebruiker gebruiker) {
        this.gebruiker = gebruiker;
    }

}
```

Bod

```
package entities;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.ManyToOne;
import javax.persistence.Table;

@Entity
@Table(name="Bod")
public class Bod {

    private Integer idBod;
    private Integer prijs;
    private String datum;
    private Advertentie advertentie;
    private Gebruiker gebruiker;

    public Bod() {
    }

    public Bod(Integer prijs, String datum, Advertentie advertentie,
```

```
                Gebruiker gebruiker){
        this();
        this.prijs = prijs;
        this.datum = datum;
        this.advertentie = advertentie;
        this.gebruiker = gebruiker;
    }

    @Id
    @GeneratedValue
    @Column(name="idBod")
    public Integer getIdBod() {
        return idBod;
    }

    public void setIdBod(Integer idBod) {
        this.idBod = idBod;
    }

    @Column(name="prijs")
    public Integer getPrijs() {
        return prijs;
    }

    public void setPrijs(Integer prijs) {
        this.prijs = prijs;
    }

    @Column(name="datum")
    public String getDatum() {
        return datum;
    }

    public void setDatum(String datum) {
        this.datum = datum;
    }

    @ManyToOne (cascade = CascadeType.ALL)
    public Advertentie getAdvertentie() {
        return advertentie;
    }

    public void setAdvertentie(Advertentie advertentie) {
        this.advertentie = advertentie;
    }

    @ManyToOne (cascade = CascadeType.ALL)
    public Gebruiker getGebruiker() {
        return gebruiker;
    }

    public void setGebruiker(Gebruiker gebruiker) {
        this.gebruiker = gebruiker;
    }
}
```

Categorie

```
package entities;

import java.util.HashSet;
import java.util.Set;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import javax.persistence.Table;

@Entity
@Table(name="categorie")
public class Categorie {
```

```
    private Integer idCategorie;
    private String categorieNaam;
    private Categorie parentCategorie;
    private Set<Categorie> sub = new HashSet<Categorie>();

    public Categorie(String categorieNaam) {
        this();
        this.categorieNaam = categorieNaam;
    }

    public Categorie() {
    }

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name="idCategorie")
    public Integer getIdCategorie() {
        return idCategorie;
    }

    public void setIdCategorie(Integer idCategorie) {
        this.idCategorie = idCategorie;
    }

    @Column(name="categorieNaam")
    public String getCategorieNaam() {
        return categorieNaam;
    }

    public void setCategorieNaam(String categorieNaam) {
        this.categorieNaam = categorieNaam;
    }

    @ManyToOne(cascade = {CascadeType.ALL})
    @JoinColumn(name="parentCategorie")
    public Categorie getParentCategorie() {
        return parentCategorie;
    }

    public void setParentCategorie(Categorie parentCategorie) {
        this.parentCategorie = parentCategorie;
    }

    @OneToMany(mappedBy = "parentCategorie" ,fetch =FetchType.EAGER)
    public Set<Categorie> getSub() {
        return sub;
    }

    public void setSub(Set<Categorie> sub) {
        this.sub = sub;
    }
}
```

Gebruiker

```
package entities;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToOne;
import javax.persistence.Table;

@Entity
@Table(name="Gebruiker")
public class Gebruiker {
    private Integer idGebruiker;
    private String voornaam;
    private String achternaam;
    private String email;
    private String wachtwoord;
    private BetalingsGegevens betalingsGegevens;

    public Gebruiker() {
```

```
    }

    public Gebruiker(String voornaam, String achternaam, String email,
            String wachtwoord) {
        this();
        this.voornaam = voornaam;
        this.achternaam = achternaam;
        this.email = email;
        this.wachtwoord = wachtwoord;
    }

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "idGebruiker")
    public Integer getIdGebruiker() {
        return idGebruiker;
    }

    public void setIdGebruiker(Integer idGebruiker) {
        this.idGebruiker = idGebruiker;
    }

    @Column(name="voornaam", nullable = false, length = 100)
    public String getVoornaam() {
        return voornaam;
    }

    public void setVoornaam(String voornaam) {
        this.voornaam = voornaam;
    }

    @Column(name="achternaam", nullable = false , length = 100)
    public String getAchternaam() {
        return achternaam;
    }

    public void setAchternaam(String achternaam) {
        this.achternaam = achternaam;
    }

    @Column(name="email", nullable = false, length = 100, unique=true)
    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    @Column(name="wachtwoord", nullable = false, length = 25)
    public String getWachtwoord() {
        return wachtwoord;
    }

    public void setWachtwoord(String wachtwoord) {
        this.wachtwoord = wachtwoord;
    }

    @OneToOne(cascade = CascadeType.ALL,mappedBy="gebruiker")
    public BetalingsGegevens getBetalingsGegevens() {
        return betalingsGegevens;
    }

    public void setBetalingsGegevens(BetalingsGegevens betalingsGegevens) {
        this.betalingsGegevens = betalingsGegevens;
    }
}
```

Scenario’s

Hieronder vindt u de code van de scenario’s. in scenario een maken we een gebruiker en een categorie aan. Hierna maken we een advertentie aan die gelinkt is aan de gebruiker en categorie.

In scenario twee maken we drie gebruikers aan. Deze drie gebruikers maken allemaal een bod op de advertentie van scenario een.

In scenario drie maken we een subcategorie gelinkt aan de categorie gemaakt in scenario een. Hierna wordt er de subcategorie gelinkt aan een advertentie.

In de laatste scenario laden we vier huidige gebruikers is. we maken een nieuwe categorie en subcategorie. We linken de subcategorie aan de categorie. Hierna maken we een advertentie gelinkt aan de eerste gebruiker (van de 4), en aan de net gemaakte categorie. Uiteindelijk plaatsten de drie overgebleven gebruikers elk een reactie op gemaakte advertentie.

```
import entities.*;

import org.hibernate.HibernateException;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;
import org.hibernate.dialect.PostgreSQLDialect;

public class App {
    private static SessionFactory factory;
    public static void main(String[] args) {
         try{
            factory = new Configuration().configure().buildSessionFactory();
          }catch (Throwable ex) {
            System.err.println("Failed to create sessionFactory object." + ex);
            throw new ExceptionInInitializerError(ex);
          }
        Session session = HibernateUtil.getSessionFactory().getCurrentSession();
        session.beginTransaction();

        App ME = new App();

        session.getTransaction().commit();

        ME.actie1();// Voer scenario 1
        ME.actie2();// Voer scenario 2
        ME.actie3();// Voer scenario 3
        ME.actie4();// Voer scenario 4

        HibernateUtil.getSessionFactory().close();
    }

    //Scenario 1 maak gebruiker en dat gebruiker maakt een advertentie
    public void actie1(){
        Session session = factory.openSession();
        Transaction tx = null;
        try {
            tx = session.beginTransaction();
            Gebruiker gebruiker = new Gebruiker("hoye", "lam", "bla@bla.nl", "1234");
            Categorie categorie = new Categorie("SmartPhones");
            Advertentie advertentie1 = new Advertentie("iPhone", "mooi", 78, true, "22-06-2015", gebruiker,
categorie);
            session.save(advertentie1);
            session.save(categorie);
            session.save(gebruiker);
            tx.commit();
        } catch (HibernateException e) {
            if (tx != null) {
                e.printStackTrace();
            }
        } finally{
            session.close();
        }
    }
```

```
//Scenario 2 meerdere gebruikers bieden op de advertentie
public void actie2(){
    Session session = factory.openSession();
    Transaction tx = null;
    try {
        tx = session.beginTransaction();
        Advertentie advertentie1 = (Advertentie) session.load(Advertentie.class, 1);
        Gebruiker gebruiker2 = new Gebruiker ("Rinesh" , "Ramadhin", "Rinesh@rinshesh.nl", "1234");
        Gebruiker gebruiker3 = new Gebruiker("Ho Ye", "lam", "bla123@bla.nl", "1234");
        Gebruiker gebruiker4 = new Gebruiker("John", "Mata", "bla21412@bla.nl", "1234");

        Bod bod1 = new Bod(70,"21-06-2015",advertentie1, gebruiker2);
        Bod bod2 = new Bod(80,"22-06-2015",advertentie1, gebruiker3);
        Bod bod3 = new Bod(50,"23-06-2015",advertentie1, gebruiker4);

        session.save(gebruiker2);
        session.save(gebruiker3);
        session.save(gebruiker4);
        session.save(bod1);
        session.save(bod2);
        session.save(bod3);

        tx.commit();
        actie2Betaal(gebruiker2);


    } catch (HibernateException e) {
        if (tx != null) {
            e.printStackTrace();
        }
    } finally{
        session.close();
    }
}

public void actie2Betaal(Gebruiker gebruiker){
    Session session = factory.openSession();
    Transaction tx = null;
    try {
        tx = session.beginTransaction();
        BetalingsGegevens betalingsGegevens = new BetalingsGegevens(451464161, "Rinesh", gebruiker);
        session.save(betalingsGegevens);
        tx.commit();
    } catch (HibernateException e) {
        if (tx != null) {
            e.printStackTrace();
        }
    } finally{
        session.close();
    }
}

public void actie3(){
    Session session = factory.openSession();
    Transaction tx = null;
    try {
        tx = session.beginTransaction();
        Categorie categorie = (Categorie) session.load(Categorie.class, 1);
        Categorie subCategorie = new Categorie("Apple");
        Advertentie advertentie1 = (Advertentie) session.load(Advertentie.class, 3);
        subCategorie.setParentCategorie(categorie);
        session.save(subCategorie);
        tx.commit();
    } catch (HibernateException e) {
        if (tx != null) {
            e.printStackTrace();
        }
    } finally{
        session.close();
    }
}

public void actie4(){
    Session session = factory.openSession();
    Transaction tx = null;
    try {
        tx = session.beginTransaction();
        Gebruiker gebruiker = (Gebruiker) session.load(Gebruiker.class, 1);
        Gebruiker gebruiker2 = (Gebruiker) session.load(Gebruiker.class, 2);
        Gebruiker gebruiker3 = (Gebruiker) session.load(Gebruiker.class, 3);
        Gebruiker gebruiker4 = (Gebruiker) session.load(Gebruiker.class, 4);

        Categorie categorie= new Categorie("Laptop");
        Categorie subCategorie = new Categorie("MSI");
```

```
            subCategorie.setParentCategorie(categorie);
            Advertentie advertentie2 = new Advertentie("MSI Laptop", "Goede staat", 500, true, "24-06-2015",
gebruiker, categorie);

            AdvertentieReactie advertentieReactie1 = new AdvertentieReactie("meer informatie graag", "26-06-2014",
advertentie2, gebruiker2);
            AdvertentieReactie advertentieReactie2 = new AdvertentieReactie("Is het beeldscherm mooi genoeg?", "26-
06-2014", advertentie2, gebruiker3);
            AdvertentieReactie advertentieReactie3 = new AdvertentieReactie("Is het beschadigd ergens?", "26-06-
2014", advertentie2, gebruiker4);

            session.save(subCategorie);
            session.save(advertentieReactie1);
            session.save(advertentieReactie2);
            session.save(advertentieReactie3);
            tx.commit();

        } catch (HibernateException e) {
            if (tx != null) {
                e.printStackTrace();
            }
        } finally{
            session.close();
        }
    }
}
```