**Министерство науки и высшего образования РФ**

**Государственное образовательное учреждение высшего профессионального образования**

**Белгородский Государственный Технологический Университет им В. Г. Шухова**

Кафедра программного обеспечения вычислительной техники и

автоматизированных систем.

Лабораторная работа №4 по дисциплине

«Технологии Web-программирования»

Тема: Разработка и проектирование базы данных

Выполнила: ст. группы ПВ-41  
Ярыгина И. Ю.  
Проверил: Картамышев С.В.

Белгород 2020 г.

**Цель работы:** изучить основы взаимодействия web-приложения с базой данных. Спроектировать базу данных для хранения информации приложения (страницы, пользователи и т.п.) и создать модели данных из базы данных.

**Выполнение**

Предполагается следующая структура приложения:

Users – таблица с пользователями, она имеет связь многие ко многим с таблицей Authority(роли), предполагается что будет 2 роли: Админ и Пользователь.

Comments – таблица с комментариями, имеет внешний ключ на пользователей и на фильм. Связь с пользователем – один ко многим (один пользователь – много комментариев), связь с фильмом - один ко многим (один фильм – много комментариев).

Films – таблица фильма, имеет внешних ключ на info\_film, связь с информацией о фильме – один к одному.

Info\_films – таблица с информацией для конкретного фильма.

Director и Genre – таблицы с именем режиссера и названием жанра соответственно, имеют связь многие ко многим с таблицей info\_film, так как фильм может принадлежать к нескольким жанрам, и к конкретным жанрам могут принадлежать разные фильмы, соответственно с таблицей режиссер ситуация аналогичная.

Итого получается следующая структура бд:

Изображение выглядит как текст, черный, снимок экрана

Автоматически созданное описание

Для реализации данной структуры необходимо создать миграции и подключиться к базе данных.

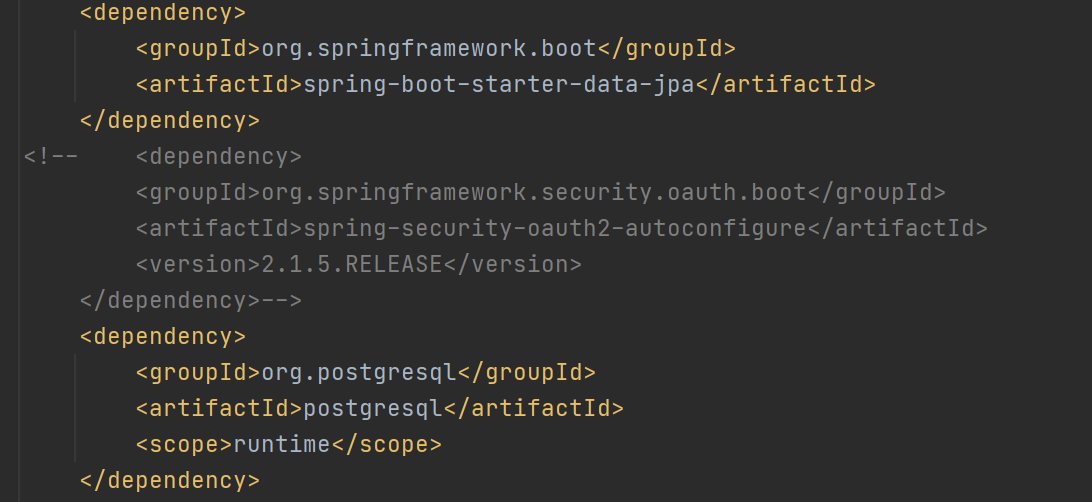
Для подключения к бд необходимо создать файл конфигурацию с название

“application” и расширением yaml или properties.

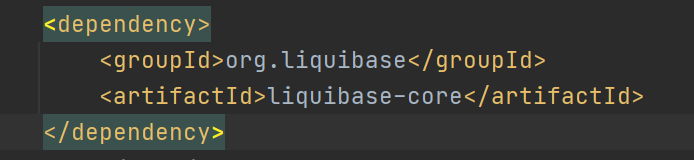
Изображение выглядит как текст

Автоматически созданное описание

Для работы с базой данных используется технология JPA.

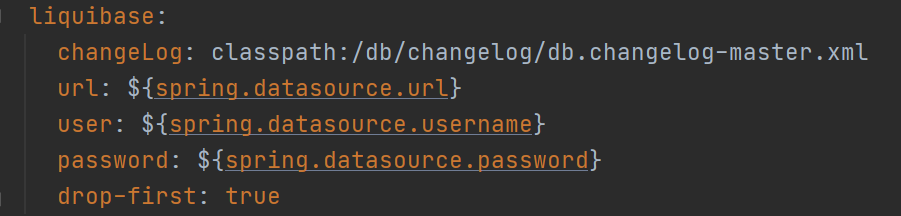


Для миграций используется библиотека Liquibase.   
Для ее подключения необходимо добавить зависимость и плагин в pom.xml

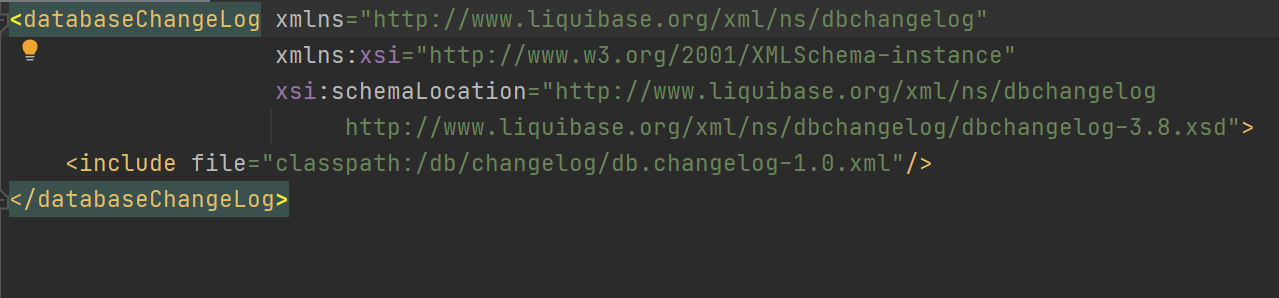




Так же необходимо указать главный xml файл, с помощью которого будут выполнены миграции. Файл указывает в application.yaml.



Главный xml файл для выполнения миграции выглядит следующим образом



Здесь через тег <include> подключаем файлы xml или sql в которых будут прописываться создаваемые таблицы для бд.

Миграции были описаны с помощью xml и выглядят следующим образом:

<databaseChangeLog xmlns="http://www.liquibase.org/xml/ns/dbchangelog"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog  
 http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.4.xsd">  
  
 <property name="blob\_type" value="bytea" dbms="postgresql"/>  
  
 <changeSet author="irina" id="17122020-2251">  
 <createTable tableName="authority">  
 <column name="id" remarks="ID" type="integer" autoIncrement="true">  
 <constraints primaryKey="true" primaryKeyName="authority\_pkey"/>  
 </column>  
 <column name="name" type="varchar(255)">  
 <constraints unique="true" uniqueConstraintName="authority\_name\_uk"/>  
 </column>  
 <column name="description" type="varchar(1024)"/>  
 <column name="date\_time\_create" remarks="Дата время вставки записи" type="datetime"/>  
 <column name="date\_time\_modif" remarks="Дата время модификации записи" type="datetime"/>  
 </createTable>  
  
 <createTable tableName="oauth\_client\_details">  
 <column name="client\_id" type="varchar(256)">  
 <constraints primaryKey="true" primaryKeyName="pk\_oauth\_client\_details"/>  
 </column>  
 <column name="resource\_ids" type="varchar(256)"/>  
 <column name="client\_secret" type="varchar(256)"/>  
 <column name="scope" type="varchar(256)"/>  
 <column name="authorized\_grant\_types" type="varchar(256)"/>  
 <column name="web\_server\_redirect\_uri" type="varchar(256)"/>  
 <column name="authorities" type="varchar(256)"/>  
 <column name="access\_token\_validity" type="int"/>  
 <column name="refresh\_token\_validity" type="int"/>  
 <column name="additional\_information" type="varchar(4096)"/>  
 <column name="autoapprove" type="varchar(256)"/>  
 </createTable>  
  
 <createTable tableName="oauth\_refresh\_token">  
 <column name="token\_id" type="varchar(255)"/>  
 <column name="token" type="${blob\_type}"/>  
 <column name="authentication" type="${blob\_type}"/>  
 </createTable>  
  
 <createTable tableName="users">  
 <column name="id" type="uuid">  
 <constraints primaryKey="true" primaryKeyName="user\_data\_pkey" unique="true" nullable="false"/>  
 </column>  
 <column name="username" type="varchar(1024)">  
 <constraints unique="true" nullable="false"/>  
 </column>  
 <column name="first\_name" type="varchar(40)">  
 <constraints nullable="false"/>  
 </column>  
 <column name="second\_name" type="varchar(40)">  
 <constraints nullable="false"/>  
 </column>  
 <column name="patronymic" type="varchar(40)">  
 <constraints nullable="false"/>  
 </column>  
 <column name="gender" type="smallint">  
 <constraints nullable="false"/>  
 </column>  
 <column name="phone" type="varchar(32)"/>  
 <column name="account\_confirmed" type="boolean"/>  
 <column name="account\_expired" type="boolean"/>  
 <column name="account\_locked" type="boolean"/>  
 <column name="credentials\_expired" type="boolean"/>  
 <column name="enabled" type="boolean"/>  
 <column name="user\_password" type="varchar(255)"/>  
 <column name="date\_time\_create" remarks="Дата время вставки записи" type="datetime">  
 <constraints nullable="false"/>  
 </column>  
 <column name="date\_time\_modif" remarks="Дата время модификации записи" type="datetime">  
 <constraints nullable="false"/>  
 </column>  
 </createTable>  
  
 <createTable tableName="user2authority">  
 <column name="user\_id" type="uuid">  
 <constraints nullable="false" primaryKey="true" primaryKeyName="user2authority\_pkey"  
 references="users(id)" foreignKeyName="user2authority\_user\_fk"  
 />  
 </column>  
 <column name="authority\_id" type="bigint">  
 <constraints nullable="false" primaryKey="true" primaryKeyName="user2authority\_pkey"  
 references="authority(id)" foreignKeyName="user2authority\_authority\_fk"/>  
 </column>  
 </createTable>  
  
 <createTable tableName="genre">  
 <column name="id" remarks="ID" type="bigint" autoIncrement="true">  
 <constraints primaryKey="true" primaryKeyName="genre\_pkey"/>  
 </column>  
  
 <column name="name" type="varchar(100)">  
 <constraints nullable="false"/>  
 </column>  
  
 </createTable>  
  
 <createTable tableName="director">  
 <column name="id" remarks="ID" type="bigint" autoIncrement="true">  
 <constraints primaryKey="true" primaryKeyName="director\_pkey"/>  
 </column>  
 <column name="name" type="varchar(100)">  
 <constraints nullable="false"/>  
 </column>  
 </createTable>  
  
 <createTable tableName="info\_film">  
 <column name="id" remarks="ID" type="bigint" autoIncrement="true">  
 <constraints primaryKey="true" primaryKeyName="info\_films\_pkey"/>  
 </column>  
 <column name="date" remarks="дата выхода фильма" type="date">  
 <constraints nullable="false"/>  
 </column>  
 </createTable>  
  
  
 <createTable tableName="films">  
 <column name="id" remarks="ID" type="bigint" autoIncrement="true">  
 <constraints primaryKey="true" primaryKeyName="films\_pkey"/>  
 </column>  
 <column name="uri\_image" type="varchar(255)">  
 <constraints nullable="false"/>  
 </column>  
  
 <column name="name" type="varchar(50)">  
 <constraints nullable="false"/>  
 </column>  
 <column name="description" type="varchar(1024)">  
 <constraints nullable="false"/>  
 </column>  
  
 <column name="info\_id" type="bigint">  
 <constraints nullable="false"  
 references="info\_film(id)"  
 foreignKeyName="films\_info\_films\_fk"  
 unique="true"/>  
 </column>  
 </createTable>  
  
  
 <createTable tableName="info2genre">  
 <column name="info\_id" type="bigint">  
 <constraints nullable="false"  
 primaryKey="true"  
 primaryKeyName="info2genre\_pkey"  
 references="info\_film(id)"  
 foreignKeyName="info\_films2genre\_fk"  
 />  
 </column>  
 <column name="genre\_id" type="bigint">  
 <constraints nullable="false"  
 primaryKey="true"  
 primaryKeyName="info2genre\_pkey"  
 references="genre(id)"  
 foreignKeyName="genre\_films2genre\_fk"  
 />  
 </column>  
 </createTable>  
  
 <createTable tableName="info2director">  
 <column name="info\_id" type="bigint">  
 <constraints nullable="false"  
 primaryKey="true"  
 primaryKeyName="info2director\_pkey"  
 references="info\_film(id)"  
 foreignKeyName="info\_info2director\_fk"  
 />  
 </column>  
 <column name="director\_id" type="bigint">  
 <constraints nullable="false"  
 primaryKey="true"  
 primaryKeyName="info2director\_pkey"  
 references="director(id)"  
 foreignKeyName="director\_info\_info2director\_fk"  
 />  
 </column>  
 </createTable>  
  
 <createTable tableName="comments">  
 <column name="id" remarks="ID" type="bigint" autoIncrement="true">  
 <constraints primaryKey="true" primaryKeyName="comments\_pkey"/>  
 </column>  
  
 <column name="date\_time\_create" remarks="Дата время вставки записи" type="datetime">  
 <constraints nullable="false"/>  
 </column>  
 <column name="date\_time\_modif" remarks="Дата время модификации записи" type="datetime">  
 <constraints nullable="false"/>  
 </column>  
  
 <column name="comment" type="varchar(100)">  
 <constraints nullable="false"/>  
 </column>  
  
 <column name="author\_id" type="uuid">  
 <constraints nullable="false" references="users(id)" foreignKeyName="comments\_users\_fk"/>  
 </column>  
  
 <column name="films\_id" type="bigint">  
 <constraints references="films(id)" foreignKeyName="films\_comment\_fk"/>  
 </column>  
  
 </createTable>  
  
  
 </changeSet>  
  
</databaseChangeLog>

Далее были добавлены сущности(отображения таблиц в виде объектов)

Изображение выглядит как текст

Автоматически созданное описание

ParentEntity.java

@Getter  
@Setter  
@MappedSuperclass  
@FieldDefaults(level = AccessLevel.*PROTECTED*)  
public class ParentEntity<T extends Serializable> {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 T id;  
}

UserEntity.java

@Entity  
@Table(name = "users")  
@Getter  
@Setter  
@EqualsAndHashCode(exclude = {"comments"})  
@NoArgsConstructor  
@AllArgsConstructor  
@Builder  
@ToString(exclude = {"comments"})  
public class UserEntity implements UserDetails {  
 @Id  
 @GeneratedValue(generator = "UserUUIDGenerator")  
 @GenericGenerator(name = "UserUUIDGenerator", strategy = "org.hibernate.id.UUIDGenerator")  
 private UUID id;  
  
 @Email  
 @NotEmpty  
 @Column(name = "username")  
 private String username;  
  
 @Column(name = "phone")  
 private String phone;  
  
 @JsonProperty(access = JsonProperty.Access.*WRITE\_ONLY*)  
 @NotEmpty  
 @Column(name = "user\_password")  
 private String password;  
  
 @Column(name = "account\_confirmed")  
 private boolean accountConfirmed;  
  
 @Column(name = "account\_expired")  
 private boolean accountExpired;  
  
 @Column(name = "account\_locked")  
 private boolean accountLocked;  
  
 @Column(name = "credentials\_expired")  
 private boolean credentialsExpired;  
  
 @Column(name = "enabled")  
 private boolean enabled;  
  
 @Column(name = "first\_name")  
 private String firstName;  
  
 @Column(name = "second\_name")  
 private String secondName;  
  
 @Column(name = "patronymic")  
 private String patronymic;  
  
 @Column(name = "date\_time\_create")  
 @CreatedDate  
 private LocalDateTime dateTimeCreate;  
  
 @Column(name = "date\_time\_modif")  
 @LastModifiedDate  
 private LocalDateTime dateTimeModif;  
  
 @OneToMany(mappedBy = "author", cascade = CascadeType.*ALL*, orphanRemoval = true)  
 private List<CommentEntity> comments;  
  
  
 @ManyToMany(fetch = FetchType.*EAGER*)  
 @JoinTable(name = "user2authority",  
 joinColumns = @JoinColumn(name = "user\_id", referencedColumnName = "id"),  
 inverseJoinColumns = @JoinColumn(name = "authority\_id",  
 referencedColumnName = "id"))  
 @OrderBy  
 private Set<AuthorityEntity> userAuthorities;  
  
 @Override  
 public Collection<? extends GrantedAuthority> getAuthorities() {  
 return null;  
 }  
  
  
 @Override  
 @JsonIgnore  
 public boolean isAccountNonExpired() {  
 return !isAccountExpired();  
 }  
  
 @Override  
 @JsonIgnore  
 public boolean isAccountNonLocked() {  
 return !isAccountLocked();  
 }  
  
 @Override  
 @JsonIgnore  
 public boolean isCredentialsNonExpired() {  
 return !isCredentialsExpired();  
 }  
  
 public void addAuthority(AuthorityEntity authority) {  
 if (userAuthorities == null) {  
 userAuthorities = new HashSet<>();  
 }  
 userAuthorities.add(authority);  
 }  
}

ModelEntity.java

@Getter  
@Setter  
@MappedSuperclass  
@FieldDefaults(level = AccessLevel.*PROTECTED*)  
public class ModelEntity <T extends Serializable> extends ParentEntity<T> {  
 @NotBlank  
 String name;  
}

InfoFilmsEntity.java

@Getter  
@Setter  
@ToString  
@EqualsAndHashCode(callSuper = true)  
@Entity(name = "info\_film")  
@Builder  
@AllArgsConstructor  
@NoArgsConstructor  
public class InfoFilmsEntity extends ParentEntity<Long>{  
  
 @Column(name = "date")  
 private LocalDate date;

@ManyToMany(fetch = FetchType.*EAGER*)  
 @JoinTable(name = "info2genre",  
 joinColumns = @JoinColumn(name = "info\_id", referencedColumnName = "id"),  
 inverseJoinColumns = @JoinColumn(name = "genre\_id", referencedColumnName = "id"))  
 private List<GenreEntity> genre;  
  
 @ManyToMany(fetch = FetchType.*EAGER*)  
 @JoinTable(name = "info2director",  
 joinColumns = @JoinColumn(name = "info\_id", referencedColumnName = "id"),  
 inverseJoinColumns = @JoinColumn(name = "director\_id", referencedColumnName = "id"))  
 private List<DirectorEntity> directors;  
}

GenryEntity.java

@Getter  
@Setter  
@ToString  
@EqualsAndHashCode(callSuper = true)  
@Entity(name = "genre")  
@Builder  
@AllArgsConstructor  
@NoArgsConstructor  
public class GenreEntity extends ModelEntity<Long> {  
 @ManyToMany(mappedBy = "genre")  
 private List<InfoFilmsEntity> info;  
}

FilmEntity.java

@Getter  
@Setter  
@ToString  
@EqualsAndHashCode(callSuper = true)  
@Entity(name = "genre")  
@Builder  
@AllArgsConstructor  
@NoArgsConstructor  
public class GenreEntity extends ModelEntity<Long> {  
 @ManyToMany(mappedBy = "genre")  
 private List<InfoFilmsEntity> info;  
}

DirectorEntity.java

@Getter  
@Setter  
@ToString  
@EqualsAndHashCode(callSuper = true)  
@Entity(name = "director")  
@Builder  
@AllArgsConstructor  
@NoArgsConstructor  
public class DirectorEntity extends ModelEntity<Long>{  
 @ManyToMany(mappedBy = "directors")  
 private List<InfoFilmsEntity> info;  
}

CommentEntity.java

@Getter  
@Setter  
@ToString  
@EqualsAndHashCode(callSuper = true)  
@Entity(name = "comments")  
@Builder  
@AllArgsConstructor  
@NoArgsConstructor  
public class CommentEntity extends ParentEntity<Long>{  
  
 @Column  
 private String comment;  
  
 @Column(name = "date\_time\_create")  
 @CreatedDate  
 private LocalDateTime dateTimeCreate;  
  
 @Column(name = "date\_time\_modif")  
 @LastModifiedDate  
 private LocalDateTime dateTimeModif;  
  
 @ManyToOne  
 @JoinColumn(name = "author\_id", referencedColumnName = "id")  
 private UserEntity author;  
  
 @ManyToOne  
 @JoinColumn(name = "films\_id", referencedColumnName = "id")  
 private FilmEntity film;  
  
}

AuthorityEntity.java

@Entity  
@Getter  
@Setter  
@Table(name = "authority")  
@AllArgsConstructor  
@NoArgsConstructor  
@Builder  
@ToString  
public class AuthorityEntity extends ParentEntity<Integer> implements GrantedAuthority {  
  
 @NotBlank  
 private String name;  
  
 private String description;  
  
 @Override  
 public String getAuthority() {  
 return getName();  
 }  
}

**Вывод:** в результате выполнения работы повторены основы проектирования бд а так же создания миграции и отображения таблиц в программе с помощью технологии ORM.