Datenbanksysteme 2, 4. Übung
ActiveRecord-Klassen

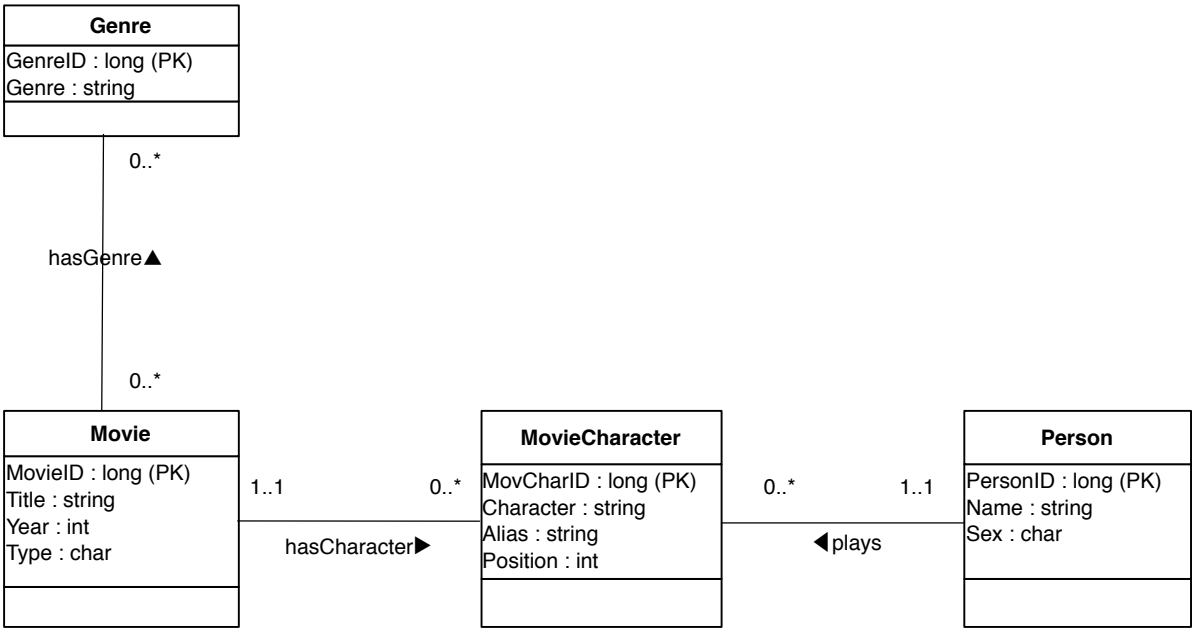
Generelles zu dieser Übung

Diese Übung sollten Sie in Gruppen von max. 4 Studierenden bearbeiten. Das Ergebnis dieser Übung wird im nächsten Übungsblatt erweitert und führt somit auf die erste Abgabe mit Bonuspunkten hin. Für das nächste Übungsblatt werden Sie 2 Wochen Zeit haben, so dass Sie ggf. dieses Übungsblatt auch dann noch fertigstellen können, falls die Zeit in dieser Woche nicht ganz reicht.

Achtung: Für das Übungsblatt muss relativ viel Code geschrieben werden, da viele (sehr ähnliche) ActiveRecord-Klassen implementiert werden müssen. Daher ist eine sinnvolle Aufteilung der Arbeit in der Gruppe wichtig, um einen vernünftigen Zeitrahmen einzuhalten!

Aufgabe 4.1 Erstellung von Entitätsklassen

Es wird folgendes Klassendiagramm verwendet:



a) Erstellen Sie ein relationales Datenmodell passend zu diesem konzeptionellen Datenmodell und legen Sie die passenden Tabellen in der Oracle-Datenbank an. Gucken Sie ggf. im Skript von DBS1 nach, wie Sie z.B. die N:M-Beziehung zwischen Movie und Genre anlegen und wie Sie die anderen Beziehungen durch Fremdschlüssel abbilden. Speichern Sie ihr SQL-Skript zum anlegen der Tabellen und ergänzen Sie passende DROP-Anweisungen, so dass Sie jederzeit das Schema neu erzeugen können. Insgesamt müssen 5 Tabellen angelegt werden.

b) Erstellen Sie für die Entitäten „Genre“, „Movie“, „MovieCharacter“ und „Person“ jeweils eine Java-Klasse mit allen Attributen und passenden get- und set-Methoden. Implementieren Sie zunächst die Method insert() für alle Klassen. Erstellen Sie

ebenso eine Klasse „MovieGenre“ für die Join-Tabelle der Assoziation zwischen Movie und Genre. Erstellen Sie dann auch die update()- und delete()-Methoden.

Beachten Sie folgendes:

1. Implementieren Sie ein Verfahren mit Hilfe von Sequenzen zum Erzeugen einer neuen ID in der Insert-Methode – allerdings nur wenn nötig: für MovieGenre ist der Primärschlüssel eine Kombination der Fremdschlüssel.
2. Die Fremdschlüssel im Datenmodell werden wie normale Attribute behandelt.
3. Teilen Sie sich die Aufgabe in der Gruppe auf!

c) Testen Sie Ihre Klassen mit folgender Methode. Prüfen Sie anschließend über den SQL Developer, ob die Daten korrekt eingefügt wurden.

```
public static void testInsert() throws SQLException {
	boolean ok = false;
	try {
		Person person = new Person();
		person.setName("Karl Tester");
		person.setSex('M');
		person.insert();

		Movie movie = new Movie();
		movie.setTitle("Die tolle Komoedie");
		movie.setYear(2012);
		movie.setType('C');
		movie.insert();

		MovieCharacter chr = new MovieCharacter();
		chr.setMovieId(movie.getId());
		chr.setPlayerId(person.getId());
		chr.setCharacter("Hauptrolle");
		chr.setAlias(null);
		chr.setPos(1);
		chr.insert();

		Genre genre = new Genre();
		genre.setGenre("Unklar");
		genre.insert();

		MovieGenre movieGenre = new MovieGenre();
		movieGenre.setGenreId(genre.getId());
		movieGenre.setMovieId(movie.getId());
		movieGenre.insert();

		DBConnection.getConnection().commit();
		ok = true;
	} finally {
		if (!ok)
			DBConnection.getConnection().rollback();
	}
}
```

d) Implementieren Sie jetzt eine Factory für Movie: MovieFactory mit den beiden Methoden

```
public Movie findById(long id);
public List<Movie> findByTitle(String title);
```

Testen Sie die Methoden, indem Sie ggf. weitere Filme einfügen und nach ihnen suchen.