Instituto Superior Técnico

Universidade Técnica de Lisboa



Information Systems and Database

Nº 77875 Filippo Campagnaro

Nº 56606 Ricardo Almeida

Nº 70553 Rui Santiago

In this project we were supposed to build an Entity-Relation Model, a SQL database, queries and a HTML/PHP Web-based information system according to a given relational model. The given relational model describes is a slightly modified and simplified solution to the scenario described in the first part of the project.

2 Conceptual model: Entity-Relation

In this section we made a ER model about the given relational model.

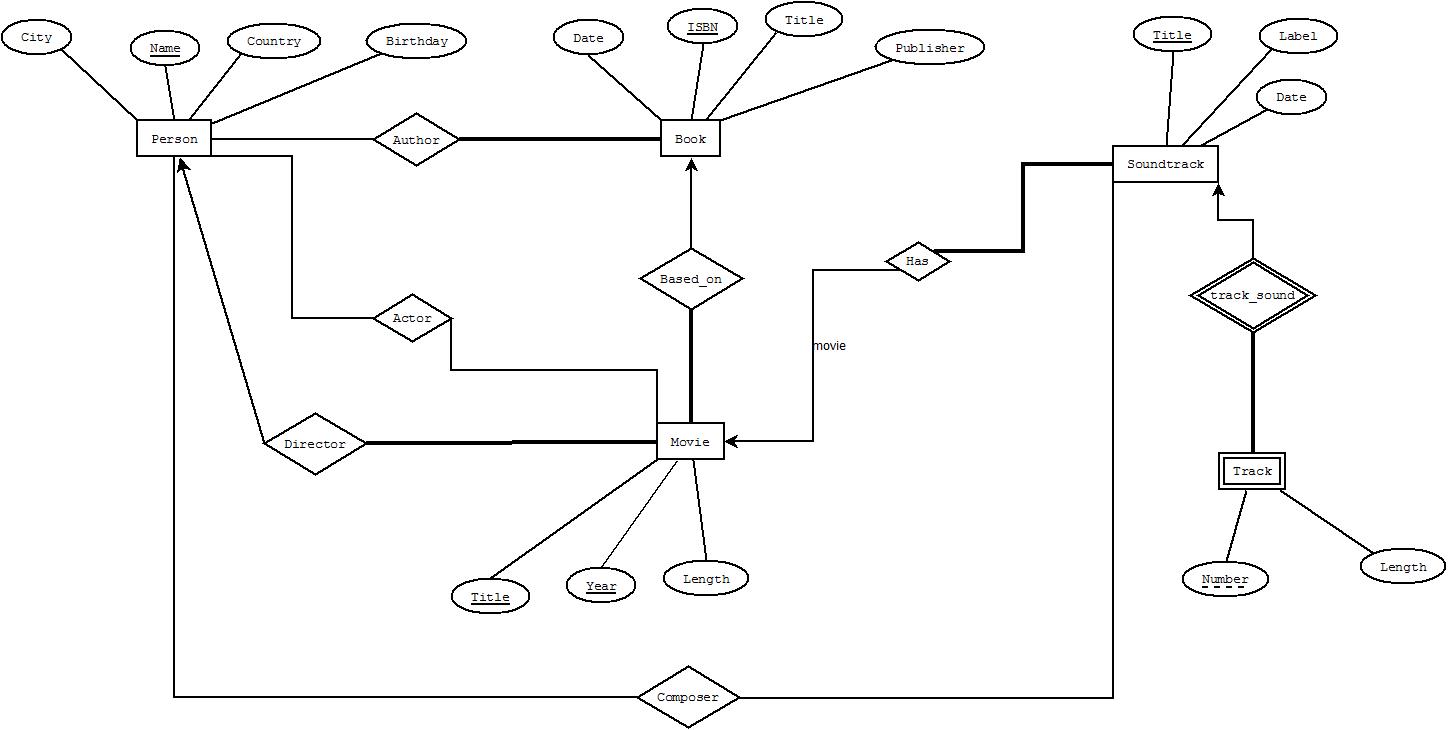


Figure 1 – ER model.

2 Database in SQL

In this section we have the Database in SQL that was designed according to the relational model given.

drop table if exists composer;

drop table if exists track;

drop table if exists soundtrack;

drop table if exists author;

drop table if exists actor;

drop table if exists movie;

drop table if exists person;

drop table if exists book;

create table person

(city varchar(255),

name varchar(255),

country varchar(255),

birthday date,

primary key(name));

create table book

(book\_date date,

isbn numeric(20,0),

title varchar(255),

publisher varchar(255),

primary key(isbn));

create table movie

(title varchar(255),

movie\_year numeric(20,0),

movie\_length numeric(20,0),

isbn numeric(20,0),

director varchar(255),

primary key(title, movie\_year),

foreign key(isbn) references book ON UPDATE CASCADE ON DELETE CASCADE,

foreign key(director) references person(name) ON UPDATE CASCADE ON DELETE CASCADE);

create table soundtrack

(title varchar(255),

label varchar(255),

soundtrack\_date date,

movie\_title varchar(255),

movie\_year numeric(20,0),

primary key(title),

foreign key(movie\_title, movie\_year) references movie(title, movie\_year) ON UPDATE CASCADE ON DELETE CASCADE);

create table track

(title varchar(255),

track\_number numeric(20,0),

trackLength numeric(20,0),

primary key(title, track\_number),

foreign key(title) references soundtrack ON UPDATE CASCADE ON DELETE CASCADE);

create table author

(isbn numeric(20,0),

name varchar(255),

primary key(isbn, name),

foreign key(isbn) references book ON UPDATE CASCADE ON DELETE CASCADE,

foreign key(name) references person ON UPDATE CASCADE ON DELETE CASCADE);

create table actor

(title varchar(255),

movie\_year numeric(20,0),

name varchar(255),

primary key(title, movie\_year, name),

foreign key(title, movie\_year) references movie ON UPDATE CASCADE ON DELETE CASCADE,

foreign key(name) references person ON UPDATE CASCADE ON DELETE CASCADE);

create table composer

(title varchar(255),

name varchar (255),

primary key(title, name),

foreign key(title) references soundtrack ON UPDATE CASCADE ON DELETE CASCADE,

foreign key(name) references person ON UPDATE CASCADE ON DELETE CASCADE);

INSERT INTO book values('1906/07/01',1,'Sporting o nosso grande amor','SportingSAD');

INSERT INTO book values('1955/05/04',2,'LORD OF THE RINGS','GRANDIEDITORI');

INSERT INTO book values('1996/05/15',3,'Harry Potter','RolEdit');

INSERT INTO person values('Lisboa','Capitolino','Portugal','1999/01/01');

INSERT INTO person values('Yate','Rowling','England','1965/07/31');

INSERT INTO person values('Bloemfontein','Tolkien','England','1892/09/02');

INSERT INTO person values('Paris','Emma Watson','England','1990/04/15');

INSERT INTO movie values('The Sporting',1992,60,1,'Capitolino');

INSERT INTO movie values('LoR',2001,60,2,'Tolkien');

INSERT INTO movie values('HP',2001,60,3,'Rowling');

INSERT INTO soundtrack values('Musica horrivel','label\_horrivel','1906/07/01','The Sporting',1992);

INSERT INTO soundtrack values('Musica hp','label\_hp','1996/07/01','HP',2001);

INSERT INTO soundtrack values('Musica LR','label\_LR','1992/07/01','LoR',2001);

INSERT INTO track values('Musica horrivel',2,20);

INSERT INTO track values('Musica horrivel',3,12);

INSERT INTO track values('Musica horrivel',4,10);

INSERT INTO author values(1,'Capitolino');

INSERT INTO actor values('The Sporting',1992,'Capitolino');

INSERT INTO actor values('HP',2001,'Emma Watson');

INSERT INTO composer values('Musica horrivel','Capitolino');

INSERT INTO person values('Treviso','Filippo','Italy','1990/09/02');

INSERT INTO person values('Lisbon','Ricardo','Portugal','1987/04/15');

INSERT INTO actor values('HP',2001,'Ricardo');

INSERT INTO actor values('HP',2001,'Filippo');

We changed variable names date to book\_date in book and date to soundtrack\_date in soundtrack because is good practice have attribute’s name different of type attributes. We also put ON UPDATE CASCADE ON DELETE CASCADE to maintain the integrity of the database.

3 SQL Queries

The requested SQL queries were solved in this section

1. SELECT name

FROM composer NATURAL JOIN (SELECT title, SUM(trackLength) AS soundtracklength

FROM soundtrack NATURAL JOIN track

GROUP BY title) AS tmp

WHERE soundtrackLength>= ALL (SELECT SUM(trackLength) AS soundtrackLength FROM soundtrack NATURAL JOIN track

GROUP BY title);

1. SELECT name FROM author NATURAL JOIN movie NATURAL JOIN actor;
2. SELECT DISTINCT director

FROM movie

WHERE director NOT IN(SELECT director

FROM person, movie NATURAL JOIN (SELECT name, title, movie\_year, country

FROM actor NATURAL JOIN person) as tmp

WHERE director=person.name AND tmp.country!=person.country ) and

director IN(select director from actor, movie where movie.title=actor.title and movie.movie\_year=actor.movie\_year);

4 Trigger

We wrote a trigger that does not allow the year of the film to be earlier than the release year of the book.

DROP TRIGGER check\_year\_movie\_trigger on movie;

DROP FUNCTION check\_year\_movie();

create function check\_year\_movie() returns trigger

as $$

begin

if new.movie\_year< (select extract(year from (select book\_date from book where new.isbn=isbn))) then

raise exception 'movie year can not be < book year.';

end if;

return new;

end

$$

language plpgsql;

create trigger check\_year\_movie\_trigger before update or insert on movie

for each row execute procedure

check\_year\_movie();

5 PHP and HTML

We made a web-based application in PHP and HTML.

a)

**pageA.php:**

<html>

<body>

<h3>Movie/Books database</h3>

<?php

$connection = pg\_connect("host=db.ist.utl.pt port=5432 user=ist177875

password=qrgt2818 dbname=ist177875") or die(pg\_last\_error());

$sql = "SELECT \* FROM movie INNER JOIN book on book.isbn=movie.isbn;";

$result = pg\_query($sql) or die(pg\_last\_error());

echo("<table border=\"0\" cellspacing=\"5\"><tr>");

for ($i=0;$i<pg\_num\_fields($result);$i++){

echo("<td> ".pg\_field\_name($result,$i)." </td>");

}

echo("</tr>");

while ($row = pg\_fetch\_row($result)) {

echo("<tr>");

for ($i=0;$i<pg\_num\_fields($result);$i++){

echo("<td>$row[$i]</td>");

}

echo("</tr>");

}

echo("</table>");

$result = pg\_free\_result($result) or die(pg\_last\_error());

pg\_close($connection);

?>

<a href='pageB.php'>second exercise</a><br/>

<a href='pageC.php'>third exercise</a><br/>

</body>

</html>

This PHP file lists all the movies in the database together with an existing book.



Figure 2 – Illustrative screenshot of pageA.php.

b)

**pageB.php:**

<html>

<body>

<h3>EsB</h3>

<form action="insertMovie.php" method='post'>

movie title:<input type="text" name="title" value=""\>

year:<input type="text" name="movie\_year" value=""\>

length:<input type="text" name="length" value=""\>

<br/>

<?php

$connection = pg\_connect("host=db.ist.utl.pt port=5432 user=ist177875

password=qrgt2818 dbname=ist177875") or die(pg\_last\_error());

$sql = "SELECT name FROM person;";

echo "<select name='director'>";

$result = pg\_query($sql) or die(pg\_last\_error());

while ($row = pg\_fetch\_assoc($result))

{

$name=$row['name'];

echo("<option value='$name'>$name</option>");

}

echo "</select><br/>";

$sql = "SELECT \* FROM book;";

$result = pg\_free\_result($result) or die(pg\_last\_error());

$result = pg\_query($sql) or die(pg\_last\_error());

while ($row = pg\_fetch\_assoc($result))

{

$isbn= $row['isbn'];

$testo=$row['title'];//" ".$row['book\_date']." ".$row['publisher'];

echo "<input type='radio' name='isbn' value='$isbn'> $testo<br/>";

}

$result = pg\_query($sql) or die(pg\_last\_error());

pg\_close($connection);

?>

<input type="submit"/>

</form>

<a href='pageA.php'>first exercise</a><br/>

<a href='pageC.php'>third exercise</a><br/>

</body>

</html>

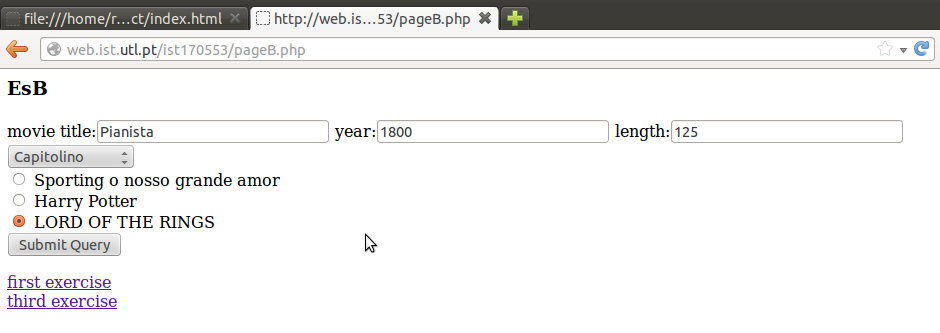


Figure 3 – Illustrative screenshot of pageB.php.

**insertMovie.php**

<html>

<body>

<h3>EsB</h3>

<?php

$title=$\_REQUEST['title'];

$movie\_year=$\_REQUEST['movie\_year'];

$length=$\_REQUEST['length'];

$director=$\_REQUEST['director'];

$isbn=$\_REQUEST['isbn'];

$connection = pg\_connect("host=db.ist.utl.pt port=5432 user=ist177875

password=qrgt2818 dbname=ist177875") or die(pg\_last\_error());

$sql = "INSERT INTO movie values('$title',$movie\_year,$length,$isbn,'$director');";

echo($sql);

$result = pg\_query($sql) or die(pg\_last\_error());

pg\_close($connection);

echo("it work");

?>

<br/>

<a href='pageA.php'>first exercise</a><br/>

<a href='pageB.php'>second exercise</a><br/>

<a href='pageC.php'>third exercise</a><br/>

</body>

</html>

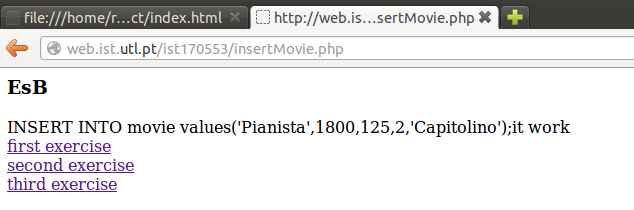


Figure 4 – Illustrative screenshot of insertMovie.php.

**pageC.php**

<html>

<body>

<h3>EsC</h3>

<form action="deleteMovie.php" method='get'>

<?php

$connection = pg\_connect("host=db.ist.utl.pt port=5432 user=ist177875

password=qrgt2818 dbname=ist177875") or die(pg\_last\_error());

$sql = "SELECT title, movie\_year FROM movie;";

$result = pg\_query($sql) or die(pg\_last\_error());

while ($row = pg\_fetch\_assoc($result))

{

$title=$row['title'];

$year=$row['movie\_year'];

echo("<input type='checkbox' name='title\_year[]' value='$title:$year'>$title $year<br/>");

}

$result = pg\_free\_result($result) or die(pg\_last\_error());

$result = pg\_query($sql) or die(pg\_last\_error());

pg\_close($connection);

?>

<input type="submit"/>

</form>

<a href='pageA.php'>first exercise</a><br/>

<a href='pageC.php'>third exercise</a><br/>

</body>

</html>

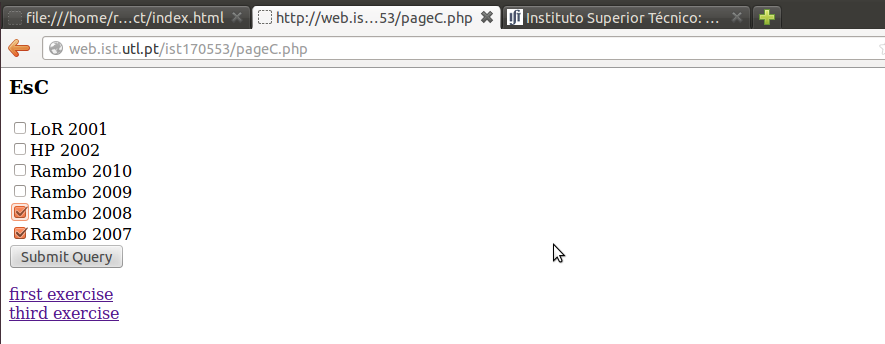
****

Figure 5 – Illustrative screenshot of pageC.php.

**deleteMovie.php**

<html>

<body>

<h3>EsB</h3>

<?php

$connection = pg\_connect("host=db.ist.utl.pt port=5432 user=ist177875

password=qrgt2818 dbname=ist177875") or die(pg\_last\_error());

echo $\_REQUEST['title\_year'];

foreach($\_REQUEST['title\_year'] as $row){

$tok=strtok($row,":");

$title=$tok;

$tok=strtok(":");

$year=$tok;

$sql = "DELETE FROM movie WHERE title='$title' and movie\_year=$year;";

echo($sql);

$result = pg\_query($sql) or die(pg\_last\_error());

}

pg\_close($connection);

echo("it work");

?>

<br/>

<a href='pageA.php'>first exercise</a><br/>

<a href='pageB.php'>second exercise</a><br/>

<a href='pageC.php'>third exercise</a><br/>

</body>

</html>



Figure 6 – Illustrative screenshot of deleteMovie.php.

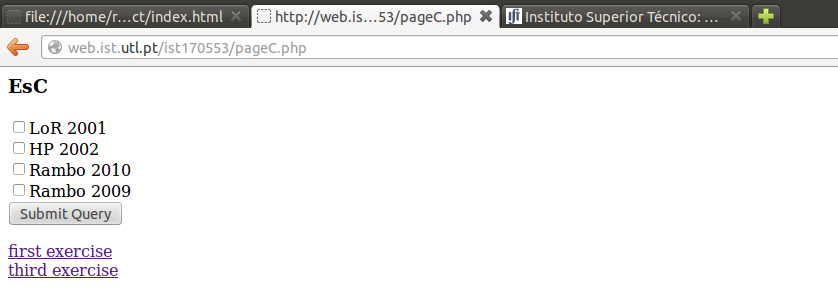


Figure 7 – Movies in database after deleting Rambo2008 and Rambo 2007