Duale Hochschule Baden-Württemberg Mannheim

Project documentation

Databases

Course of Studies: Wirtschaftsinformatik

Branch Data Science

Authors:	Hannah Schult, Sofie Pischl, Jana Lugeder
matriculation number	5373022, 3943911, 7807816
course:	WWI20DSB
course administrator:	Petko Rutesic
date of submission:	15.07.2022

Table of Contents

1 Requirements specification

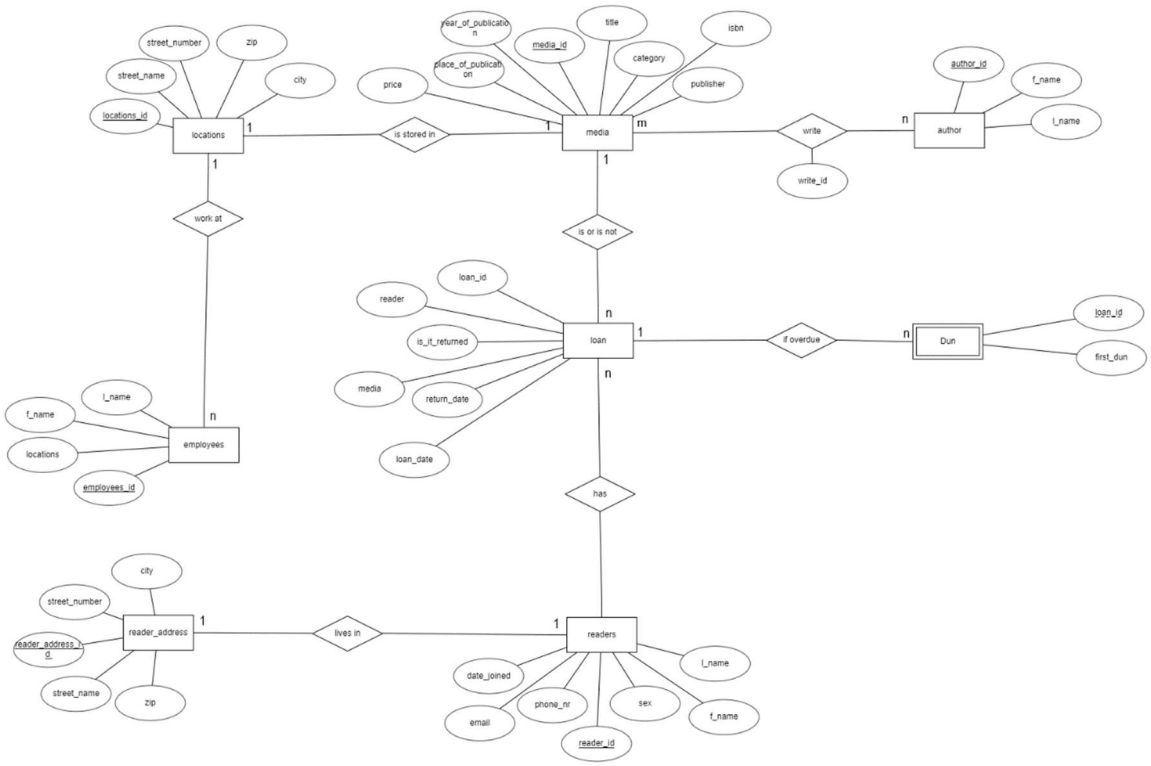
In the course of the project, we noticed that our specification had to be adapted a bit, which is why individual tables and attributes have changed. In addition, we sometimes had to add foreign keys

Problem: We create a database to organize multiple library locations.

Problem Specification:

- The library keeps track of media, loan, reader, dun, authors, rel media author employees, location and reader address.
- Each media is described by its title, category, ISBN, publisher, place of publication, year of publication, price, location and a unique media ID. The information about the author is outsourced to the author table and between the two tables there is the mapping table rel media author.
- Each author is described by it's first and last name
- Each loan is described by its loan ID, media ID, reader ID, loan date, return date and a is it returned boolean. Loan ID, media ID and reader ID are unique values.
- Each reader has a first name, last name, phone number, email, sex, id and a data joined value. The readers address is outsourced to the reader address table.
- Each reader address is defined by street, street number, zip, city and its ID.
- Each dun has loan and first dun, so it's a weak entity.
- Every Employee is described by first name, last name, his or her location and a unique ID.
- The differently located libraries are described by their unique location ID street name, street number, zip and city.

2 ER Diagram


street_number
zip
street_name
city
locations_id
locations
1
1
is stored in
price
year_of_publicatio
n
media_id
title
isbn
place_of_publicati
on
category
publisher
1
media
m
write
n
author
author_id
f_name
l_name
write_id
1
work at
is or is not
loan_id
reader
is_it_returned
media
return_date
loan_date
n
loan
1
if overdue
n
Dun
loan_id
first_dun
n
l_name
f_name
locations
employees_id
employees
has
city
street_number
reader_address_i
d
street_name
zip
reader_address
1
lives in
1
readers
l_name
date_joined
phone_nr
sex
f_name
email
reader_id

3 Transforming ER-diagram into Relational Model

3.1 Analyse for 3NF

A relation is in the third normal form (3NF) if it is in the second normal form (2NF) and all non-prime attributes directly (non-transitively) dependent on keys and not on other non-prime attributes.

This is given, because all our tables have an “id” column, which can be used to access all other attributes in the table. This was achieved by splitting all the tables to the form, in that every other attribute can be identified by the id. There is no other attribute with dependencies except the id. For example, the last name of an employee depends on the employee_id and so does the location of the employee. Location does not depend on the first or last name and none of the other attributes of other tables depend on a not key attribute.

The second normal form is given, because, as explained before, every attribute is dependent on the respective id. The second condition of 2NF is the first normal form, which is given if the tables only contain atomic values (not more than one value) and there are no repeating groups, which means that there must not be the same value in different columns.

3.2 DDL statements and insert commands

3.2.1 DDL Statements

```
CREATE TABLE reader_address(
reader_address_id serial UNIQUE NOT NULL,
primary key(reader_address_id),
street_name character varying(300) NOT NULL,
street_number character varying(4) NOT NULL,
zip character varying(20),
city character varying(40)
);

CREATE TABLE readers(
readers_id serial UNIQUE NOT NULL,
primary key(readers_id),
f_name character varying(200) NOT NULL,
l_name character varying(200) NOT NULL,
phone_nr character varying(20),
email character varying(300),
sex character varying (1),
reader_address integer not null,
foreign key (reader_address) references
reader_address(reader_address_id),
date_joined date NOT NULL DEFAULT current_date
);

CREATE TABLE locations(
locations_id serial UNIQUE NOT NULL,
primary key(locations_id),
street_name character varying(300) NOT NULL,
street_number character varying(4) NOT NULL,
zip character varying(20),
city character varying(40)
);

CREATE TABLE media(
media_id serial UNIQUE NOT NULL,
primary key(media_id),
title character varying(200) NOT NULL,
category character varying(100) NOT NULL,
ISBN character varying(60) NOT NULL,
publisher character varying (100) NOT NULL,
place_of_publication character varying(200) NOT NULL,
year_of_publication character varying(20) NOT NULL,
price numeric NOT NULL,
locations integer not null,
foreign key (locations) references locations(locations_id)
);
```

```
CREATE TABLE loan(
loan_id serial UNIQUE NOT NULL,
primary key(loan_id),
reader integer not null,
media integer not null,
is_it_returned boolean not null,
foreign key (reader) references readers(readers_id),
foreign key (media) references media(media_id),
loan_date date not null default current_date,
return_date date not null default current_date+7
);

CREATE TABLE dun(
loan integer,
foreign key (loan) references loan(loan_id),
first_dun date not null default current_date+8
);

CREATE TABLE author(
author_id serial UNIQUE NOT NULL,
primary key(author_id),
f_name character varying(200) NOT NULL,
l_name character varying(200) NOT NULL
);

CREATE TABLE rel_media_author(
media integer,
author integer,
foreign key (media) references media(media_id),
foreign key (author) references author(author_id)
);

CREATE TABLE employees(
employees_id serial UNIQUE NOT NULL,
primary key(employees_id),
f_name character varying(200) NOT NULL,
l_name character varying(200) NOT NULL,
locations integer,
foreign key (locations) references locations(locations_id)
)
```

3.2.1 Insert Statements

```
Insert into reader_address (street_name, street_number, zip, city)
values
('Musterstraße', '67', '69115', 'Mannheim'),
('Feldweg', '13', '69115', 'Heidelberg'),
('Hauptstaße', '152', '69115', 'Mannheim'),
('Rosenweg', '12', '69115', 'Mannheim'),
('Kaiser-Wilhelm-Straße', '5', '67061', 'Ludwigshafen'),
('Schulweg', '24', '69115', 'Mannheim'),
('Schulweg', '24', '69115', 'Mannheim');

Insert into readers (f_name, l_name, phone_nr, email, sex,
reader_address)
values
('Lea', 'Müller', 0643276348, 'lea.müller@gmx.de', 'f', 1),
('Tobias', 'Kraußmann', 06432862458, 'tobias.kraussmann@gmail.com', 'm',
2),
('Harald', 'Job', 0623145289, 'h.job@gmx.de', 'm', 3),
('Sarah', 'Deutsch', 06231649802, 'deutsch.sarah@googlemail.com', 'f',
4),
('Peter', 'Hohl', 06543348233, 'peter.hohl@gmail.com', 'm', 5),
('Markus', 'Meier', 0654352437, 'meier.markus@gmx.de', 'm', 6),
('Johanna', 'Meier', 0654352437, 'meier.johanna@gmx.de', 'f', 7);

Insert into locations (street_name, street_number, zip, city)
values
('Holunderbütenstraße', '4', '68159', 'Mannheim' ),
('Hauptstraße', '132', '68159', 'Mannheim' ),
('Feldweg', '24', '69115', 'Heidelberg' );

insert into media (title, category, isbn, publisher,
place_of_publication, year_of_publication, price, locations)
values
('Freiheit für alle: Das Ende der Arbeit wie wir sie kannten',
'Fachbücher für Sozialwissenschaft', '3442315514', 'Goldmann Verlag',
'NaN', '2022', 24.00, 1),
('oxen. Noctis', 'Thriller', '3423263245', 'dtv Verlagsgesellschaft mbH
& Co. KG', 'NaN', '2022', 16.95, 1),
('Die Rezeptur', 'Thriller', '3944676351', 'Kafel Verlag', 'NaN',
'2022', 10.99, 1),
('Der böse Mann', 'Thriller', '3944676319', 'Kafel Verlag','NaN',
'2021', 10.99, 2),
('Gesunde Ernährung 3 in 1 Kochbuch', 'Ratgeber', '979-8689228419',
'Independently published', 'NaN', '2020', 14.95, 2),
('Die Paradiese von gestern', 'Roman', '3963116145', 'Mitteldeutscher
Verlag', 'NaN', '2022', 28.00, 2),
('Single Bald Female', 'Roman', '152907424X', 'Macmillan', 'NaN',
'2022', 18.79, 1),
```

```
('Kein Wort mehr über Liebe', 'Roman', '3499008319', 'Rowohlt
Taschenbuch', 'NaN', '2021', 12.00, 3),
('Die Anomalie', 'Thriller', '-', 'Rowohlt', 'NaN', '2021', 22.00, 3 ),
('Der Salzpfad', 'Roman', '-', 'DUMONT REISEVERLAG', 'NaN', '2019',
10.00, 3 ),
('Die Betrogene', 'Krimi', '978-3-7341-0085-7', 'Blanvalet', 'NaN',
'2015', 10.99, 1),
('Das Rosie-Projekt', 'Roman', '978-3-596-19700-2', 'Fischer
Taschenbuch', 'NaN', '2015', 11.00, 2 ),
('Er ist wieder da', 'Roman', '978-3-404-17178-1', 'Bastei Lübbe',
'NaN', '2014', 11.00, 3 ),
('Vielleicht mag ich dich morgen', 'Roman', '978-3-426-51647-8', 'Knaur
Taschenbuch', 'NaN', '2014', 10.99, 1 ),
('Vielleicht mag ich dich morgen', 'Roman', '978-3-426-51647-8', 'Knaur
Taschenbuch', 'NaN', '2014', 10.99, 2 ),
('Vielleicht mag ich dich morgen', 'Roman', '978-3-426-51647-8', 'Knaur
Taschenbuch', 'NaN', '2014', 10.99, 2 ),
('Die Paradiese von gestern', 'Roman', '3963116145', 'Mitteldeutscher
Verlag', 'NaN', '2022', 28.00, 3 );

Insert into author (f_name, l_name)
values
('Catherine', 'Sheperd'),
('Lidia', 'Ahlers'),
('Mario', 'Schneider'),
('Laura', 'Price'),
('Herve', 'Le Tellier'),
('Romy', 'Ritte'),
('Jürgen', 'Ritte'),
('Raynor', 'Winn'),
('Christa', 'Prummer-Lehmair'),
('Heide', 'Horn'),
('Charlotte', 'Link'),
('Graeme', 'Simsion'),
('Timur', 'Vermes'),
('Mhairi', 'McFaralane'),
('Richard David', 'Precht'),
('Jens Henrik', 'Jensen'),
('Friederike', 'Buchinger');

Insert into employees (f_name, l_name, locations)
values
('Hannah', 'Grab', 1),
('Felix', 'Schmitz', 1),
('Simon', 'Lutz', 2),
('Fabienne', 'Fußer', 2),
('Lena', 'Kurz', 3),
('Rebecca', 'Bossert', 3);

Insert into loan (reader, media, is_it_returned)
```

```
values
(4, 10, 'no'),
(3, 16, 'no'),
(5, 7, 'no'),
(1, 8, 'no');



Insert into rel_media_author (media, author)
values
(1, 15),
(2, 16),
(2, 17),
(3, 1),
(4, 1),
(5, 2),
(6, 3),
(7, 4),
(8, 5),
(8, 6),
(8, 7),
(9, 5),
(9, 6),
(9, 7),
(10, 8),
(10, 9),
(10, 10),
(11, 11),
(12, 12),
(13, 13),
(14, 14),
(15, 14),
(16, 14),
(17, 3);
```

3.3 Relational Model

media_id [PK] integer	title character varying (200)	category character varying (100)	isbn character varying (60)	publisher character varying (100)	place_of_publication character varying (200)	year_of_publication character varying (20)	price numeric	locations integer
1	Freiheit für alle: Das Ende der Arbeit wie wir sie kannten	Fachbücher für Sozialwissenschaft	3442315514	Goldmann Verlag	NaN	2022	24.00	1
2	oxen. Noctis	Thriller	3423263245	dtv Verlagsgesellschaft mbH & Co. KG	NaN	2022	16.95	1
3	Die Rezeptur	Thriller	3944676351	Kafel Verlag	NaN	2022	10.99	1
4	Der böse Mann	Thriller	3944676319	Kafel Verlag	NaN	2021	10.99	2
5	Gesunde Ernährung 3 in 1 Kochbuch	Ratgeber	979-8689228419	Independently published	NaN	2020	14.95	2
6	Die Paradiese von gestern	Roman	3963116145	Mitteldeutscher Verlag	NaN	2022	28.00	2
7	Single Bald Female	Roman	152907424X	Macmillan	NaN	2022	18.79	1
8	Kein Wort mehr über Liebe	Roman	3499008319	Rowohlt Taschenbuch	NaN	2021	12.00	3
9	Die Anomalie	Thriller	-	Rowohlt	NaN	2021	22.00	3
10	Der Salzpfad	Roman	-	DUMONT REISEVERLAG	NaN	2019	10.00	3
11	Die Betrogene	Krimi	978-3-7341-0085-7	Blanvalet	NaN	2015	10.99	1
12	Das Rosie-Projekt	Roman	978-3-596-19700-2	Fischer Taschenbuch	NaN	2015	11.00	2
13	Er ist wieder da	Roman	978-3-404-17178-1	Bastei Lübbe	NaN	2014	11.00	3
14	Vielleicht mag ich dich morgen	Roman	978-3-426-51647-8	Knaur Taschenbuch	NaN	2014	10.99	1
15	Vielleicht mag ich dich morgen	Roman	978-3-426-51647-8	Knaur Taschenbuch	NaN	2014	10.99	2
16	Vielleicht mag ich dich morgen	Roman	978-3-426-51647-8	Knaur Taschenbuch	NaN	2014	10.99	2
17	Die Paradiese von gestern	Roman	3963116145	Mitteldeutscher Verlag	NaN	2022	28.00	3

readers_id [PK] integer	f_name character varying (200)	l_name character varying (200)	phone_nr character varying (20)	email character varying (300)	sex character varying (1)	reader_address integer	date_joined date
1	Lea	Müller	643276348	lea.müller@gmx.de	f	1	2022-07-08
2	Tobias	Kraußmann	6432862458	tobias.kraussmann@gmail.com	m	2	2022-07-08
3	Harald	Job	623145289	h.job@gmx.de	m	3	2022-07-08
4	Sarah	Deutsch	6231649802	deutsch.sarah@googlemail.com	f	4	2022-07-08
5	Peter	Hohl	6543348233	peter.hohl@gmail.com	m	5	2022-07-08
6	Markus	Meier	654352437	meier.markus@gmx.de	m	6	2022-07-08
7	Johanna	Meier	654352437	meier.johanna@gmx.de	f	7	2022-07-08

author_id [PK] integer	f_name character varying (200)	l_name character varying (200)
1	Catherine	Sheperd
2	Lidia	Ahlers
3	Mario	Schneider
4	Laura	Price
5	Herve	Le Tellier
6	Romy	Ritte
7	Jürgen	Ritte
8	Raynor	Winn
9	Christa	Prummer-Lehmair
10	Heide	Horn
11	Charlotte	Link
12	Graeme	Simsion
13	Timur	Vermes
14	Mhairi	McFaralane
15	Richard David	Precht
16	Jens Henrik	Jensen
17	Friederike	Buchinger

	reader_address_id [PK] integer	street_name character varying (300)	street_number character varying (4)	zip character varying (20)	city character varying (40)
1	1	Musterstraße	67	69115	Mannheim
2	2	Feldweg	13	69115	Heidelberg
3	3	Hauptstaße	152	69115	Mannheim
4	4	Rosenweg	12	69115	Mannheim
5	5	Kaiser-Wilhelm-Straße	5	67061	Ludwigshafen
6	6	Schulweg	24	69115	Mannheim
7	7	Schulweg	24	69115	Mannheim

	loan integer	first_dun date

	employees_id [PK] integer	f_name character varying (200)	l_name character varying (200)	locations integer
1	1	Hannah	Grab	1
2	2	Felix	Schmitz	1
3	3	Simon	Lutz	2
4	4	Fabienne	Fußer	2
5	5	Lena	Kurz	3
6	6	Rebecca	Bossert	3

	loan_id [PK] integer	reader integer	media integer	is_it_returned boolean	loan_date date	return_date date
1	1	4	10	false	2022-07-08	2022-07-15
2	2	3	16	false	2022-07-08	2022-07-15
3	3	5	7	false	2022-07-08	2022-07-15
4	4	1	8	false	2022-07-08	2022-07-15

	media integer	author integer
1	1	15
2	2	16
3	2	17
4	3	1
5	4	1
6	5	2
7	6	3
8	7	4
9	8	5
10	8	6
11	8	7
12	9	5
13	9	6
14	9	7
15	10	8
16	10	9
17	10	10
18	11	11
19	12	12
20	13	13
21	14	14
22	15	14
23	16	14
24	17	3

	locations_id [PK] integer	street_name character varying (300)	street_number character varying (4)	zip character varying (20)	city character varying (40)
1	1	Holunderbütenstraße	4	68159	Mannheim
2	2	Hauptstraße	132	68159	Mannheim
3	3	Feldweg	24	69115	Heidelberg

3.4 simple views and materialized views

The first materialized view is a view to see the readers with their address.

`Reader_with_addresses`

```
create materialized view Reader_with_addresses as
SELECT readers.readers_id,
      readers.f_name,
      readers.l_name,
      readers.reader_address,
      reader_address.reader_address_id,
      reader_address.street_name,
      reader_address.street_number,
      reader_address.zip,
      reader_address.city
      FROM readers
            JOIN reader_address ON
readers.readers_id=reader_address.reader_address_id;
```

Output for: `SELECT * FROM reader_with_addresses`

Data Output | Explain | Messages | Notifications

	readers_id integer	f_name character varying (200)	l_name character varying (200)	reader_address integer	reader_address_id integer	street_name character varying (300)	street_number character varying (4)	zip character varying (20)	city character varying (40)
1	1	Lea	Müller	1	1	Musterstraße	67	69115	Mannheim
2	2	Tobias	Kraußmann	2	2	Feldweg	13	69115	Heidelberg
3	3	Harald	Job	3	3	Hauptstaße	152	69115	Mannheim
4	4	Sarah	Deutsch	4	4	Rosenweg	12	69115	Mannheim
5	5	Peter	Hohl	5	5	Kaiser-Wilhelm-Straße	5	67061	Ludwigshafen
6	6	Markus	Meier	6	6	Schulweg	24	69115	Mannheim
7	7	Johanna	Meier	7	7	Schulweg	24	69115	Mannheim

The second materialized view is a view to see all books with their authors.

```
create materialized view books_with_authors as
select med.media_id, med.title, aut.f_name, aut.l_name from media med
left join rel_media_author rel on med.media_id = rel.media
left join author aut on aut.author_id = rel.author

Output for: SELECT * from books_with_authors
```

Data Output | Explain | Messages | Notifications

	media_id integer	title character varying (200)	f_name character varying (200)	l_name character varying (200)
1	1	Freiheit für alle: Das Ende der Arbeit wie wir sie kannten	Richard David	Precht
2	2	oxen. Noctis	Jens Henrik	Jensen
3	2	oxen. Noctis	Friederike	Buchinger
4	3	Die Rezeptur	Catherine	Sheperd
5	4	Der böse Mann	Catherine	Sheperd
6	5	Gesunde Ernährung 3 in 1 Kochbuch	Lidia	Ahlers
7	6	Die Paradiese von gestern	Mario	Schneider
8	7	Single Bald Female	Laura	Price
9	8	Kein Wort mehr über Liebe	Herve	Le Tellier
10	8	Kein Wort mehr über Liebe	Romy	Ritte
11	8	Kein Wort mehr über Liebe	Jürgen	Ritte
12	9	Die Anomalie	Herve	Le Tellier
13	9	Die Anomalie	Romy	Ritte
14	9	Die Anomalie	Jürgen	Ritte
15	10	Der Salzpfad	Raynor	Winn
16	10	Der Salzpfad	Christa	Prummer-Lehmair
17	10	Der Salzpfad	Heide	Horn
18	11	Die Betrogene	Charlotte	Link
19	12	Das Rosie-Projekt	Graeme	Simsion
20	13	Er ist wieder da	Timur	Vermes
21	14	Vielleicht mag ich dich morgen	Mhairi	McFaralane
22	15	Vielleicht mag ich dich morgen	Mhairi	McFaralane
23	16	Vielleicht mag ich dich morgen	Mhairi	McFaralane
24	17	Die Paradiese von gestern	Mario	Schneider

This simple view show all borrowed books and who borrowed them:
`Borrowed_books_with_reader:`

```
create view Borrowed_books_with_reader as
SELECT loan.loan_id,
      loan.reader,
        loan.media,
        loan.is_it_returned,
        media.media_id,
        media.title,
         media.locations,
         readers.readers_id,
         readers.f_name,
         readers.l_name
        FROM loan
            JOIN media ON loan.media = media.media_id
            JOIN readers ON readers.readers_id = loan.reader
      WHERE loan.is_it_returned = false;
```

Output for: `SELECT * FROM borrowed_books_with_reader`

Data Output | Explain | Messages | Notifications

	loan_id integer	reader integer	media integer	is_it_returned boolean	media_id integer	title character varying (200)	locations integer	readers_id integer	f_name character varying (200)	l_name character varying (200)
1	1	4	10	false	10	Der Salzpfad	3	4	Sarah	Deutsch
2	2	3	16	false	16	Vielleicht mag ich dich morgen	2	3	Harald	Job
3	3	5	7	false	7	Single Bald Female	1	5	Peter	Hohl
4	4	1	8	false	8	Kein Wort mehr über Liebe	3	1	Lea	Müller

3.5 procedure

We create a procedure to delete readers from our library.

```
Create Procedure Sp_DeleteReadersById(
@ReadersId int) as
Begin
        Delete From readers where readers_id = @StudentId
End
```

3.6 SQL queries

In the following section some queries on our earlier explained database will be written down and explained. Always the query is described by the text above and the result table is printed below if meaningful.

 # show books which are currently borrowed, the respective reader and when it has to be returned

The goal of the first query written down in the below code snipped is to show a table with all books which are currently borrowed including the reader who borrows it and the date they have to return it. To reach the goal the following three tables are getting joined to each other with respectively inner joins. Additionally, a where-clause is included to filter only those media which are not already returned.

```
select med.media_id, med.title, r.f_name, r.l_name, l.return_date from
loan l
inner join media med on l.media = med.media_id
inner join readers r on l.reader = r.readers_id
where l.is_it_returned = 'false'
```

	media_id integer	title character varying (200)	f_name character varying (200)	l_name character varying (200)	return_date date
1	10	Der Salzpfad	Sarah	Deutsch	2022-07-17
2	7	Single Bald Female	Peter	Hohl	2022-07-17
3	8	Kein Wort mehr über Liebe	Lea	Müller	2022-07-17

 # Renaming: Update of a last name after marriage

Now two readers are getting married. Lea Müller and Tobias Kraußmann asked the library to change Tobias' last name into 'Müller'. He is authenticated with his library card, his personal data are getting called with his id which is '2'.

```
update readers set l_name = 'Müller' where readers_id=2;
```

 # Number of books every author has written in the inventory of the library

Next a table gets printed which contains the number of books every author has written in the complete inventory of the library. In this case the table rel_media_author is getting joined to the author table. Also, the results of this right join are getting grouped by the information about the author and the number of media gets counted. Lastly the resulting table gets ordered by the computed count of media in a descending order.

```
select rel.author, aut.f_name ||' '|| aut.l_name as author,
count(rel.media) from author aut
right join rel_media_author rel on rel.author = aut.author_id
group by rel.author, aut.f_name, aut.l_name, author
order by count desc;
```

	author integer	author text	count bigint
1	14	Mhairi McFaralane	3
2	3	Mario Schneider	2
3	1	Catherine Sheperd	2
4	7	Jürgen Ritte	2
5	6	Romy Ritte	2
6	5	Herve Le Tellier	2
7	12	Graeme Simsion	1
8	8	Raynor Winn	1
9	17	Friederike Buchinger	1
10	16	Jens Henrik Jensen	1
11	10	Heide Horn	1
12	15	Richard David Precht	1
13	13	Timur Vermes	1
14	11	Charlotte Link	1
15	4	Laura Price	1
16	2	Lidia Ahlers	1
17	9	Christa Prummer-Lehmair	1

 # Display of all authors who wrote at least one book of the inventory of the library which was written before 2020 and the respective book(s)

Furthermore a nested function will be described now. The result of the following query is a table where all authors who wrote at least one book of the inventory of the library which was written before 2020 are getting listed and their respective book(s). First, a few tables are being joined to each other: rel_media_author, media and author.

If the integer of the attribute year_of_publication from the media table is smaller than 2020 the row will be shown in the result table, otherwise not.

```
select distinct aut.f_name, aut.l_name, med.title,
med.year_of_publication from rel_media_author rel
right join media med on med.media_id = rel.media
right join author aut on aut.author_id = rel.author
where (select cast(med.year_of_publication as int) from media med where
med.media_id = rel.media) < 2020;
```

	f_name character varying (200)	l_name character varying (200)	title character varying (200)	year_of_publication character varying (20)
1	Heide	Horn	Der Salzpfad	2019
2	Charlotte	Link	Die Betrogene	2015
3	Mhairi	McFaralane	Vielleicht mag ich dich morgen	2014
4	Raynor	Winn	Der Salzpfad	2019
5	Christa	Prummer-Lehmair	Der Salzpfad	2019
6	Timur	Vermes	Er ist wieder da	2014
7	Graeme	Simsion	Das Rosie-Projekt	2015

 # Count of media in every location

In the next query the goal is to provide an overview over the amount of books in every location. To have all necessary attributes the tables locations and media are getting joined. Only some attributes which describe the location briefly and the count of the media titles are getting presented. The table gets grouped by the information of the locations so that a count of media per location is possible.

```
select loc.locations_id, loc.city, count(med.title) from locations loc
left join media med on med.locations = loc.locations_id
group by loc.locations_id, loc.city
```

	locations_id [PK] integer	city character varying (40)	count bigint
1	2	Mannheim	6
2	3	Heidelberg	5
3	1	Mannheim	6

 # List of all people having a connection to the library either as customer or as employee

Furthermore a set operation is included in the next query. To create a list of all people having a connection to the library either as customer or as employee for a circular letter about the 10th birthday of the library the last and first names of the two tables employees and readers are getting united. This means this time no attributes are getting added to a table but values are getting appended.

```
select emp.f_name, emp.l_name from employees emp
union select re.f_name, re.l_name from readers re
```

	f_name character varying (200)	l_name character varying (200)
1	Tobias	Müller
2	Markus	Meier
3	Peter	Hohl
4	Felix	Schmitz
5	Sarah	Deutsch
6	Harald	Job
7	Fabienne	Fußer
8	Simon	Lutz
9	Johanna	Meier
10	Hannah	Grab
11	Lea	Müller
12	Lena	Kurz
13	Rebecca	Bossert

 # Display of all personal information of all readers living in Mannheim

For a promotion matter the complete personal information of all readers living in Mannheim shall be shown. For this matter the readers table gets joined to the reader_address table. In the last row of the query its defined that only the rows with the value ‘Mannheim’ at the attribute ‘city’ of the table reader_address are getting considered. Lastly only relevant information without duplicates are getting selected for a better overview.

```
select radd.reader_address_id, re.f_name, re.l_name, radd.street_name, radd.street_number,
radd.city, re.phone_nr, re.email from reader_address radd
inner join readers re on re.readers_id = radd.reader_address_id
where radd.city = 'Mannheim'
```

	reader_address_id integer	f_name character varying (200)	l_name character varying (200)	street_name character varying (300)	street_number character varying (4)	city character varying (40)	phone_nr character varying (20)	email character varying (300)
1	1	Lea	Müller	Musterstraße	67	Mannheim	643276348	lea.müller@gmx.de
2	3	Harald	Job	Hauptstaße	152	Mannheim	623145289	h.job@gmx.de
3	4	Sarah	Deutsch	Rosenweg	12	Mannheim	6231649802	deutsch.sarah@googlemail.com
4	6	Markus	Meier	Schulweg	24	Mannheim	654352437	meier.markus@gmx.de
5	7	Johanna	Meier	Schulweg	24	Mannheim	654352437	meier.johanna@gmx.de

 # Publisher with the most books in the inventory of the library

The publisher with the most books in the inventory of the library is being searched for a little gift. For this reason the author with the maximum entries in the attribute 'publisher' of the media table is getting selected. For a better understanding the resulting attribute name is being changed to 'most_occuring_publisher'.

select max(med.publisher) as most_occuring_publisher from media med

	most_occuring_publisher text
1	Rowohlt Taschenbuch

 # locations where medium with the id=1 is located

The next goal is to show properly, in which locations the medium with the id=1 is located to tell a customer who is calling where he can find the respective book. The media table is being joined with the locations table and the criterium to show the rows is that the attribute media id must be '1'.

```
select med.media_id, med.title, loc.locations_id, loc.city from media
med
inner join locations loc on loc.locations_id = med.locations
where med.media_id = 1
```

	media_id integer	title character varying (200)	locations_id integer	city character varying (40)
1	1	Freiheit für alle: Das Ende der Arbeit wie wir sie kannten	1	Mannheim

 # Books written by Mhairi McFaralane, its media descriptions and where it is located

In the following all book written by Mhairi McFaralane, its media descriptions and its locations shall be shown properly. To get this result table the following tables are getting joined: author, rel_media_author and media. The whole result gets filtered by the requirement that the first name of the author must be ‘Mhairi’ and the last name ‘McFaralane’. For readable reasons not all attributes of the result tables is being displayed here.

```
select aut.author_id, aut.f_name ||' '|| aut.l_name, med.media_id,
med.title, med.category, med.isbn, med.publisher,
med.year_of_publication, med.price, med.locations from author aut
inner join rel_media_author rel on aut.author_id = rel.author
inner join media med on med.media_id = rel.media
where aut.f_name = 'Mhairi' and aut.l_name = 'McFaralane'
```

Data Output | Explain | Messages | Notifications

	author_id integer	?column? text	media_id integer	title character varying (200)	category character varying (100)	isbn character varying (60)	publisher character varying (100)	year_ chara
1	14	Mhairi McFaralane	14	Vielleicht mag ich dich morgen	Roman	978-3-426-51647-8	Knaur Taschenbuch	2014
2	14	Mhairi McFaralane	15	Vielleicht mag ich dich morgen	Roman	978-3-426-51647-8	Knaur Taschenbuch	2014
3	14	Mhairi McFaralane	16	Vielleicht mag ich dich morgen	Roman	978-3-426-51647-8	Knaur Taschenbuch	2014

 # Get amount of feminin readers

For mothers day the library wants to get an idea of the amount of feminin customers to decide wether they want to provide a little gift for all feminin customers or not. So the count of the rows of the table readers where the attribute ‘sex’ is ‘f’ is getting computed.

```
select count(*) from readers
where sex = 'f'
```

	count bigint
1	3

 # Show amount of book 'Vielleicht mag ich dich morgen' written by Mhairi McFaralane in every location

The book with the title ‘Vielleicht mag ich dich morgen’ written by Mhairi McFaralane is very often in demand. The library wants to know the amount of the respective book per location to check wether they have to buy some more which is being executed in the following query. There the following few tables are getting joined: locations, media, rel_media_author and author. Some attributes are getting selected for the result table and in particular the sum of occurences of the following values in the respective attributes are getting summed: title = ‘Vielleicht mag ich

dich morgen', author_id = 14, f_name = 'Mhairi' and l_name = 'McFaralane'. The whole table is grouped by the location.

```
select loc.locations_id, sum(case when med.title='Vielleicht mag ich
dich morgen' and aut.author_id = 14 and aut.f_name = 'Mhairi' and
aut.l_name = 'McFaralane' then 1 else 0 end) from locations loc
left join media med on loc.locations_id = med.locations
left join rel_media_author rel on med.media_id = rel.media
left join author aut on rel.author = aut.author_id
group by loc.locations_id;
```

	locations_id [PK] integer	sum bigint
1	2	2
2	3	0
3	1	1

 # Loan with id=2 is getting returned at the current day à loan table needs to get updated
Lastly, of course book which are borrowed will be returned. In the last query the book with the id=2 will be registered as returned in the loan table. The update is being executed as follows: The attribute 'is_it_returned' is being changed to 'true' and the return_date is being set to the current date for the book with the id=2. For a visual display the resulting table is being printed and ordered by loan_id.

```
-- loan with id=2 is getting returned at the current day --> loan table
needs to get updated
update loan set is_it_returned = 'true', return_date = current_date
where loan_id = 2;
select * from loan
order by loan_id;
```

	loan_id [PK] integer	reader integer	media integer	is_it_returned boolean	loan_date date	return_date date
1	1	4	10	false	2022-07-10	2022-07-17
2	2	3	16	true	2022-07-10	2022-07-11
3	3	5	7	false	2022-07-10	2022-07-17
4	4	1	8	false	2022-07-10	2022-07-17