GROUP PROJECT FINAL REPORT IN DATABASE DESIGN - GROUP 12

GROUP ID: 12
GROUP MEMBERS:
Liv Ullum S5754569
Mingkai Xu S5115159
Xiaoyu Zhou S5669685

Original User Requirements

You are asked to create a database for managing a collection for a new museum in Groningen. Every piece in the museum is classified in one of these three categories: manuscripts, paintings, and artifacts.

The museum has opening and closing times.
The museum has different employees: cashiers; janitors; researchers. Visitors may reserve their tickets.

Extended User Requirements

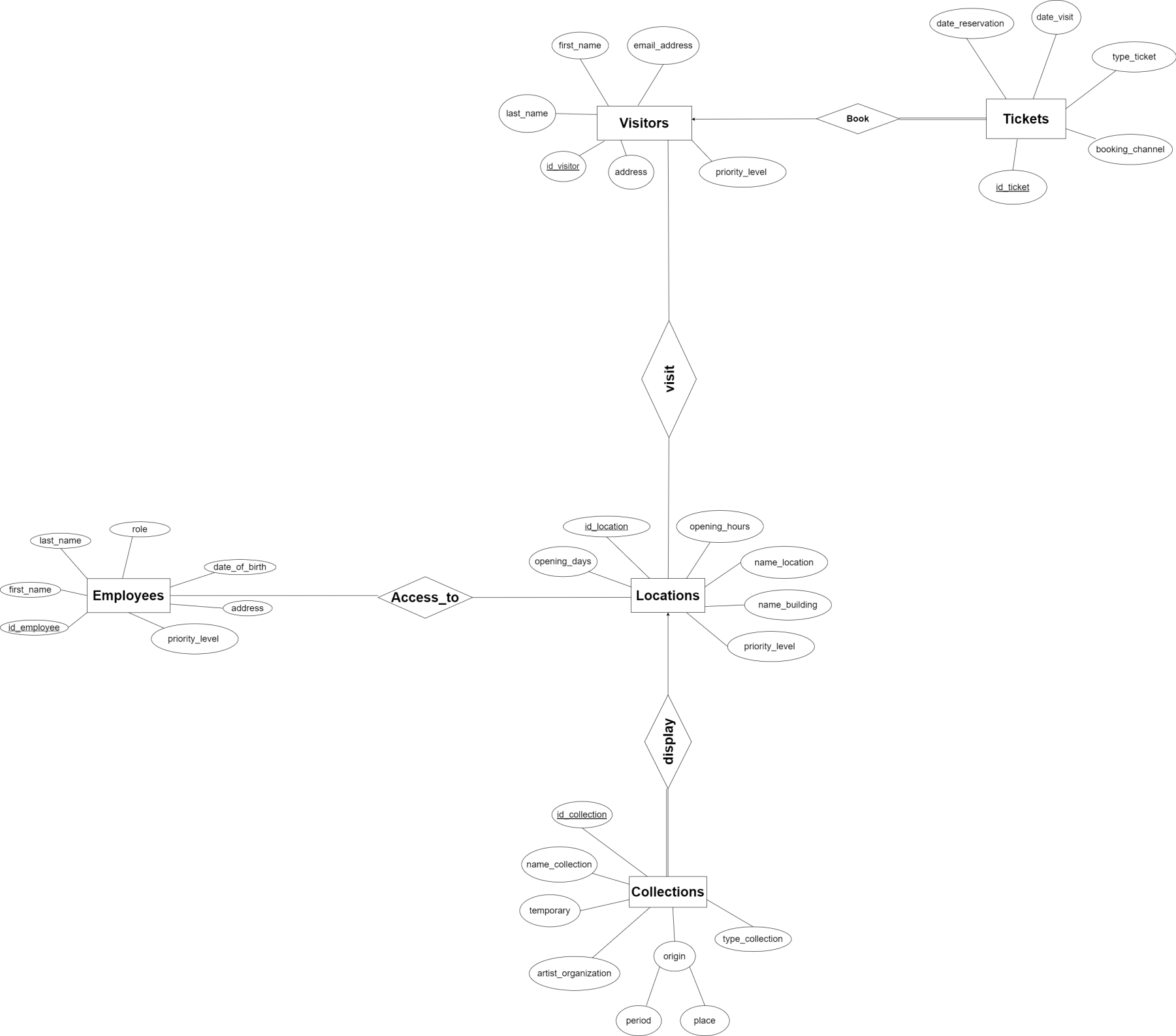
We were asked to design a database for a new museum in Groningen. In the database, we had to consider the following criteria:

The museum consists of two buildings: *Peerd van Omeloeks* and *Martini Toren* that each contain different locations. *Peerd van Omeloeks* contains two wings, a canteen, and a storage for cleaning supplies, whereas *Martini Toren* contains one wing, a storage for the art that is not displayed, and room for the administration. The administration consists of both offices and a research lab. For each location, we want to know the location ID, the name of the building it belongs to, and when it is open. The museum has three main categories of their collections: manuscripts, paintings, and artifacts. The collections all need to be defined by an ID, the name of the piece, the artist, the origin including the period and place, and if it is a temporary piece of art in the museum or if it is constantly on show. There are six different types of employees: tour guides, security guards, janitors, cashiers, researchers, and the employees working in the administration. For the employees, we need their ID, first and last name, role, date of birth, and address. In regard to the visitors, there is only one type of visitor, but we should access information about the visitor's email, address, first and last name, and they should have an ID. The visitors can reserve tickets, both online and at the museum. There are five different types of tickets: elder discount, student discount, children discount, membership card, and full price. The tickets have a ticket number and two types of dates: one for when the reservation was made and one for when they visited.

Furthermore, the database should show which locations the visitors have access to and which employees have access to which locations. This is done by adding priority levels to the employees, visitors, and locations. For the locations, the wings and the canteen have priority level 3 and are open for everyone. The administration and the cleaning supply storage has priority level 2, which means only employees with level 2 or higher can access. The storage for the art and the research lab can only be accessed by employees with priority level 1. Visitors are the only persons with priority level 3. Cahiers and tour guides have priority level 2 as they can not access the main storage and research lab. The rest of the employees, the janitors, guards, researchers and administration have priority level 1 so they can access every room in the building.

E-R DIAGRAM

The ER-diagram can be found on the next page.


first_name
email_address
last_name
Visitors
id_visitor
address
priority_level
Book
Tickets
date_reservation
date_visit
type_ticket
booking_channel
id_ticket
visit
id_location
opening_hours
opening_days
name_location
Locations
name_building
priority_level
role
last_name
date_of_birth
first_name
Employees
address
id_employee
priority_level
Access_to
display
id_collection
name_collection
Collections
temporary
type_collection
origin
artist_organization
period
place

RELATIONAL SCHEMAS

Visitors:(email_address, first_name, last_name, id_visitor, address, priority_level)

Tickets: (date_reservation, date_visit, type_ticket, id_ticket, booking_channel)

Locations: (id_location, opening_hours, name_location, name_building, priority_level, opening_days)

Collections: (type_collection, period, place, artists_organization, temporary, name_collection, id_collection)'

Employees: (id_employee last_name, first_name, role, date_of_birth, address, priority_level)

Book(id_visitor, id_ticket)

FRN KEY id_visitor REFERS TO id_visitor in Visitors

FRN KEY id_ticket REFERS TO id_ticket in Tickets

Visit(id_visitor, id_location)

FRN KEY id_visitor REFERS TO id_visitor in Visitors

FRN KEY id_location REFERS TO id_location in Locations

Display(id_collection, id_location)

FRN KEY id_location REFERS TO id_location in Locations

FRN KEY id_collection REFERS TO id_collection in Collections

Access_to (id_employee, id_location)

FRN KEY id_location REFERS TO id_location in Locations

FRN KEY id_employee REFERS TO id_employee in Employees

SQL AS DDL

```
CREATE TABLE visitors (
   id_visitor INT PRIMARY KEY NOT NULL,
   first_name VARCHAR(50) NOT NULL,
```

```
    last_name VARCHAR(50) NOT NULL,
    email_address VARCHAR(50) NOT NULL,
    address VARCHAR(200) NOT NULL,
    priority_level DEFAULT 3
);

INSERT INTO visitors (id_visitor, first_name, last_name, email_address, address)
VALUES (129087654, 'Xiaoming', 'Wang', 'huluhu@gmail.com', '98 adc street 1584');
           (354267534, 'John', 'Smith', 'johnsmith@gmail.com', '84 St main 1334'),
            (653298610, 'Emma', 'Martin', 'emmamartin@gmail.com', '67 Elm St 576')
            (145732654, 'San', 'Zhang', 'sanzhang@gmail.com', '404 Cedar 8765');
            (987126982, 'Ava', 'Lee', 'avalee@gmail.com', '797 Willow Ln 9815');

CREATE TABLE collections (
    type_collection VARCHAR(50) NOT NULL,
    period VARCHAR(50),
    place VARCHAR(50),
    artists_organization VARCHAR(50),
    temporary BOOLEAN,
    name_collection VARCHAR(50) NOT NULL,
    id_collection INT PRIMARY KEY NOT NULL
);

INSERT INTO collections (type_collection, period, place, artists_organization, temporary, name_collection, id_collection)
VALUES ('manuscript', '1770', 'Shanxi', 'Xu De', 'yes', 'Chunjianghuayueye', 20901),
           ('manuscript', '1881', 'Egypt', 'Emma Green', 'yes', 'Entertainment', 20981),
```

```
 ('artifact', '1801', 'Datong', 'Wangwei', 'yes', 'Mountains', 10090),

('artifact', '207', 'Zhaoge', 'Li Bai', 'no', 'Chill', 50711),

('painting', '1609', 'London', 'April Fisher', 'yes', 'Farming', 20090);


CREATE TABLE locations (

   id_location INT PRIMARY KEY NOT NULL,

   opening_hours TIME NOT NULL,

   opening_days VARCHAR(50) NOT NULL,

   name_location VARCHAR(50) NOT NULL,

   name_building VARCHAR(50) NOT NULL,

   priority_level DECIMAL(3, 0) NOT NULL,

   CONSTRAINT name_location CHECK (name_location IN ('East Wing', 'West Wing', 'Logistics Storage', 'Canteen', 'North Wing', 'Collection Storage', 'Administration', 'Research Lab')),

    CONSTRAINT name_building CHECK (name_building IN ('Peerd van Omeloeks', 'Martini Toren'))

);


INSERT INTO locations (id_location, opening_hours, opening_days, name_location, name_building, priority_level)

VALUES (1001, '10:00-20:00', 'Tuesday-Sunday', 'East Wing', 'Peerd van Omeloeks', 3),

 (1002, '10:00-20:00', 'Tuesday-Sunday', 'West Wing', 'Peerd van Omeloeks', 3),

(1003, '10:00-20:00', 'Tuesday-Sunday', 'Logistics Storage', 'Peerd van Omeloeks', 1),

(1004, '10:00-20:00', 'Tuesday-Sunday', 'Canteen', 'Peerd van Omeloeks', 3),

(1005, '10:00-20:00', 'Tuesday-Sunday', 'North Wing', 'Martini Toren', 3),

(1006, '10:00-20:00', 'Tuesday-Sunday', 'Collection Storage', 'Martini Toren', 1),

(1007, '10:00-20:00', 'Tuesday-Sunday', 'Administration', 'Martini Toren', 2),

(1008, '10:00-20:00', 'Tuesday-Sunday', 'Research Lab', 'Martini Toren', 1);
```

```
CREATE TABLE tickets (
    id_ticket INT PRIMARY KEY,
    date_reservation DATE NUT NULL,
    date_visit DATE NUT NULL,
    type_ticket VARCHAR(50) NUT NULL,
    booking_channel VARCHAR(50) NUT NULL
);

INSERT INTO tickets (id_ticket, date_reservation, date_visit, type_ticket, booking_channel)
VALUES (34521, '2022-09-12', '2022-12-12', 'elder discount', 'at museum'),
        (98101, '2022-02-01', '2022-02-12', 'student discount', 'online'),
        (78123, '2022-02-01', '2022-02-12', 'full price', 'online');
        (95491, '2022-02-01', '2022-02-12', 'student discount', 'online');
        (12008, '2022-08-22', '2022-08-31', 'children discount', 'online');
        (10081, '2022-09-30', '2022-10-10', 'membership discount', 'at museum');
        (18951, '2022-09-30', '2022-10-10', 'membership discount', 'at museum');
        (27001, '2022-06-08', '2022-06-28', 'full price', 'online');

CREATE TABLE employees (
  id_employee INT PRIMARY KEY NOT NULL,
  last_name VARCHAR(50) NOT NULL,
  first_name VARCHAR(50) NOT NULL,
  role VARCHAR(50) CHECK (role IN ('Tour guides', 'Security guards', 'Janitors', 'Cashiers', 'Researchers', 'Administration')),
  date_of_Birth DATE NOT NULL,
  address VARCHAR(50) NOT NULL,
  priority_level DECIMAL(1,0) NOT NULL
```

```
);


INSERT INTO employees (id_employee, last_name, first_name, role, date_of_Birth, address, priority_level)
VALUES (1423, 'Hansen', 'Lui', 'Researchers', '1971-04-23', 'Winschoterdiep 30', 1),
    (1618, 'Werner', 'Elis', 'Researchers', '1985-06-18', 'Meieweg 11', 1),
    (1313, 'Wu', 'Heya', 'Janitors', '1965-03-13', 'Europaweg 19', 1),
    (1413, 'Visser', 'Janneke', 'Administration', '1992-04-13', 'Lillaweg 5', 1),
    (1112, 'Zheng', 'Lina', 'Administration', '1976-01-12', 'Oosterport 25', 1),
    (1711, 'Halle', 'Jelmer', 'Tour guides', '1991-07-11', 'Turinstraat 46', 2),
    (1108, 'Flores', 'Lily', 'Cashiers', '2001-11-08', 'Meerweg 51', 2);


CREATE TABLE book (
    id_visitor INT,
    id_ticket INT,
    FOREIGN KEY (id_visitor) REFERENCES visitors(id_visitor),
    FOREIGN KEY (id_ticket) REFERENCES tickets(id_ticket)
);


INSERT INTO book (id_visitor, id_ticket)
VALUES (129087654, 34521),
        (354267534, 98101), (354267534, 78123), (354267534, 95491),
        (653298610, 12008),
        (145732654, 10081), (145732654, 18951),
        (987126982, 27001);


CREATE TABLE visit (
```

```
    id_visitor INT,
    id_location INT,
    FOREIGN KEY (id_visitor) REFERENCES visitors(id_visitor),
    FOREIGN KEY (id_location) REFERENCES locations(id_location)
);

INSERT INTO visit (id_visitor, id_location)
VALUES (129087654, 1001), (129087654, 1002), (129087654, 1004), (129087654, 1005),
       (354267534, 1001), (354267534, 1004), (354267534, 1005), (354267534, 1002),
       (653298610, 1004), (653298610, 1005), (653298610, 1002), (145732654, 1001),
       (145732654, 1005), (145732654, 1002), (145732654, 1001), (987126982, 1001),
       (987126982, 1002), (987126982, 1004), (987126982, 1005);

CREATE TABLE display (
   id_collection INT,
   id_location INT,
   FOREIGN KEY (id_location) REFERENCES locations (id_location),
   FOREIGN KEY (id_collection) REFERENCES collections (id_collection)
);

INSERT INTO display (id_collection, id_location)
VALUES(20901, 1001), (20981, 1002), (10090, 1006), (50711, 1005), (20090,1008);

CREATE TABLE access_to (
    id_employee INT,
    id_location INT,
```

```
    FOREIGN KEY (id_employee) REFERENCES employees(id_employee),
    FOREIGN KEY (id_location) REFERENCES locations(id_location)
);

INSERT INTO access_to (id_employee, id_location)
VALUES(1423,1001),(1423,1002), (1423,1003), (1423,1004), (1423,1005), (1423,1006), (1423,1007), (1423,1008),
  (1618,1001),(1618,1002), (1618,1003), (1618,1004), (1618,1005), (1618,1006), (1618,1007), (1618,1008),
  (1313,1001), (1313,1002), (1313,1003), (1313,1004), (1313,1005), (1313,1006), (1313,1007), (1313,1008),
  (1413,1001),(1413,1002), (1413,1003), (1413,1004), (1413,1005), (1413,1006), (1413,1007), (1413,1008),
   (1112,1001),(1112,1002), (1112,1003), (1112,1004), (1112,1005), (1112,1006), (1112,1007), (1112,1008),
    (1711,1001),(1711,1002), (1711,1004), (1711,1005), (1711,1007),
    (1108,1001),(1108,1002), (1108,1004), (1108,1005), (1108,1007);
```

Queries / SQL as DML

Who bought a student ticket?

```
SELECT book.id_visitor, tickets.type_ticket
FROM book
INNER JOIN tickets ON book.id_ticket = tickets.id_ticket
WHERE tickets.type_ticket = 'student discount';
```

Who bought their ticket online?

```
SELECT book.id_visitor, tickets.booking_channel
```

```
FROM book
INNER JOIN tickets ON book.id_ticket = tickets.id_ticket
WHERE tickets.booking_channel = ‘online’;
```

How many employees currently work there?

```
SELECT COUNT(employees.id_employee)
From Employees;
```

Make a joined table for the employees and the locations with priority level 1

```
SELECT employees.id_employee, employees.priority_level, locations.id_location
FROM Employees
INNER JOIN locations ON locations.priority_level = employees.priority_level
WHERE employees.priority_level = 1;
```

Where are the different collections displayed?

```
SELECT *
FROM collections
JOIN display ON display.id_collection = collections.id_collection;
```