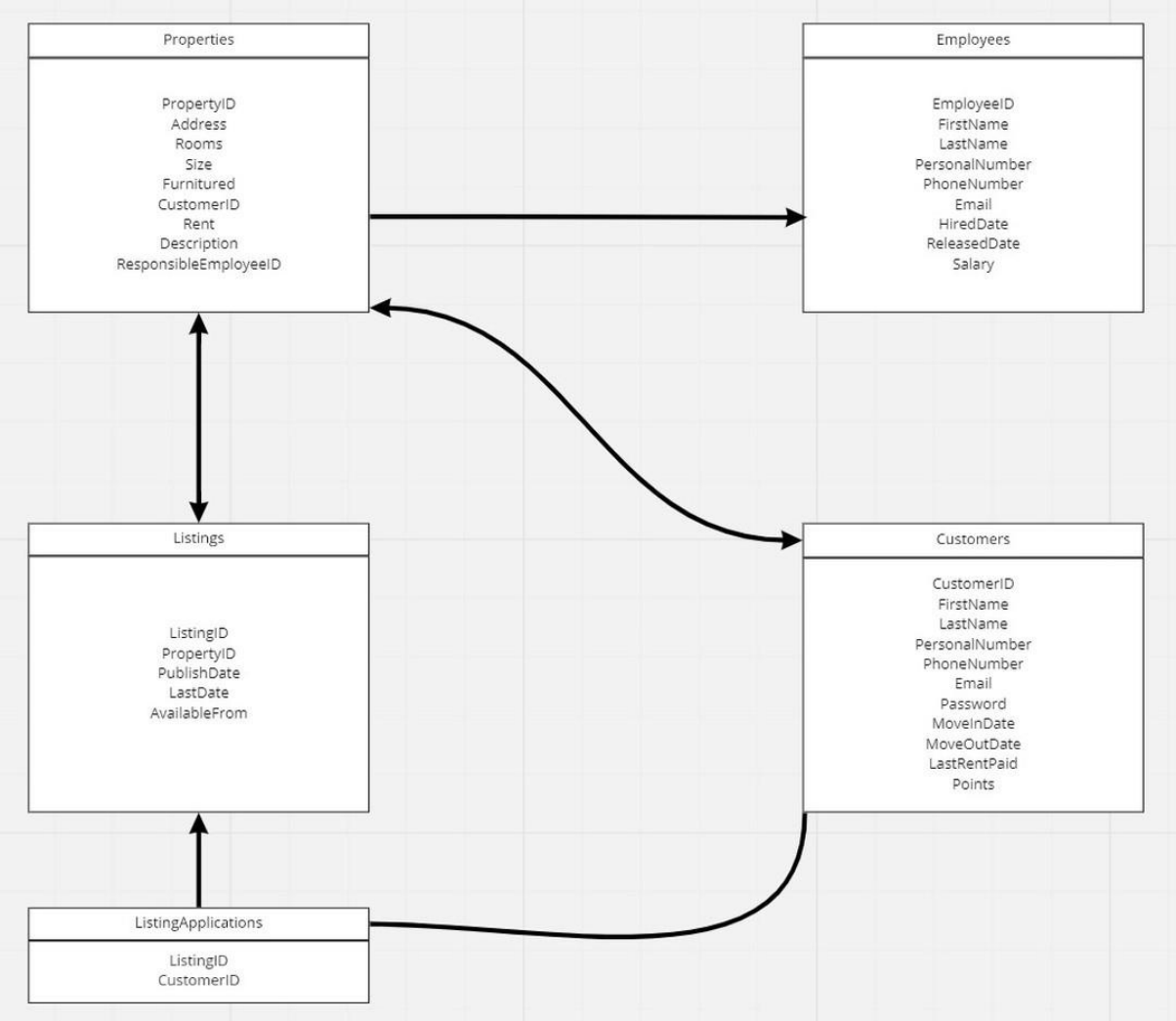
Programming Assignment 2 Report

Student(s):	Nedko Nedkov - nn222mx@student.lnu.se Tobin Tom – tt222nb@student.lnu.se

1. Project Idea

Our project idea was to implement database of a housing agency. We generated the data randomly through a data generating tool we found online.

2. Schema Design



The diagram above is our design schema. Every relation between the entity sets are “has” relations. We chose to do the schema this way for the sake of simplicity. The entity sets in the diagram are Properties, Employees, Customers, Listings and ListingApplications. The Properties set is related to Employees set through a many-one relation. The Properties set is also related to Customers set through a one-one relation and Listings set which is also a one-one relation. The Listings set is, in turn, related to the ListingApplications set through a one-many relation. Finally, the Customers set and the ListingApplications set is related to each other through a many-many relation.

3. SQL Queries

Here you present and discuss the most interesting queries. Make sure you have 5 of them at least and check the specification in the assignment sheet. One example is found below:

Q: **Get the most wanted property and the number of applicants who applied for the property.**
The query joins three tables to get the desired result. The query joins listing_applications table, listings table and properties table, and selects the properties which have applications. And then this is ordered in descending order on the count of the customerID and this is limited by 1.

```
   SELECT p.Address, COUNT(la.customerID) AS applications\
     FROM listing_applications AS la \
     JOIN listings AS l \
       ON la.listingID = l.listingID \
     JOIN properties AS p \
       ON l.propertyID = p.propertyID \
GROUP BY la.listingID \
ORDER BY COUNT(la.customerID) DESC \
   LIMIT 1;")  # Get the property for which most people have applied
```

Q: **Get the employee managing a specific property by address.**
This query shows the usage of procedure. The first name and last name of the employee is the result of this query. The address in the query is put in by the user.

```
CREATE PROCEDURE \
get_employee_in_address(IN address VARCHAR(200), OUT first_name VARCHAR(20), OUT last_name VARCHAR(20)) \
BEGIN  \
SELECT e.first_name, e.last_name \
INTO first_name, last_name \
FROM properties AS p \
JOIN employees AS e \
ON p.employeeID = e.employeeID \
WHERE p.address = address; \
END")  # Get the employee managing a specific property by address
```

Q: **Get the employee managing the property of a specific tenant by personal number.**
This query also uses PROCEDURE and, in this case, the personal number of a tenant is put in by the user. It also uses JOIN to join Properties, Employees and Customers.

```
CREATE PROCEDURE \
get_employee_by_tenant(IN personal_number CHAR(13), OUT first_name VARCHAR(20), OUT last_name VARCHAR(20))
BEGIN  \
SELECT e.first_name, e.last_name \
  INTO first_name, last_name \
  FROM properties AS p \
  JOIN employees AS e \
    ON p.employeeID = e.employeeID \
  JOIN customers AS c \
    ON p.customerID = c.customerID \
 WHERE c.personal_number = personal_number; \
END")  # Get the employee managing the property of a specific tenant by personal number
       # This query shows the usage of a procedure
```

Q: **Get all employees managing a property which a tenant is moving out of.**
VIEW is used in this query and first name and last name of the employees is the output. The query joins Properties, Employees and customers.

```
CREATE VIEW employees_losing_tenants AS \
     SELECT e.first_name, e.last_name \
       FROM properties AS p \
       JOIN employees AS e \
         ON p.EmployeeID = e.EmployeeID \
       JOIN customers AS c \
         ON p.CustomerID = c.CustomerID \
      WHERE c.move_out_date IS NOT NULL;")
```

Q: **Get the average points of all customers managed by each employee.**
This is another query which uses VIEW. This query gets the first name, last name and average points of customers of each employee.

```
CREATE VIEW average_points_by_employee AS \
     SELECT e.first_name, e.last_name, AVG(c.points) \
       FROM properties AS p \
       JOIN customers AS c \
         ON p.customerID = c.customerID \
       JOIN employees AS e \
         ON p.employeeID = e.employeeID \
   GROUP BY p.employeeID \
   ORDER BY AVG(c.points) DESC;")  # Get the average points of all
                                   # customers managed by each employee
```

Q: **Get the average salary of employees responsible for the properties the most interested customer has applied.**
Yet another query which uses VIEW. This query uses every table of the database to get the desired outcome.

```
CREATE VIEW average_salary_wanted_employees AS \
SELECT AVG(e.salary) \
FROM employees AS e \
WHERE e.employeeID IN (\
    SELECT p.employeeID \
    FROM properties AS p \
    WHERE p.propertyID IN (\
        SELECT l.propertyID \
        FROM listings AS l \
        WHERE l.listingID IN (\
            SELECT la.listingID \
            FROM listing_applications AS la \
            WHERE la.customerID = (\
                SELECT c.customerID \
                FROM listing_applications AS la \
                JOIN customers AS c \
                ON la.customerID = c.customerID \
                GROUP BY c.customerID \
                ORDER BY COUNT(la.listingID) DESC \
                LIMIT 1)));")  # AVG salaries of the employees managing the properties,
                               # for which the most interested customer has applied
                               # Query is just to show the usage of:
                               # Select, aggregate functions, where, subquery, join,
                               # group by, order by and limit all in a view
```

4. Discussion and Resources

The project uses python to code. A python module datetime is used to change a string into a date format. We also used mysql.connector to connect the python code to MySQL server. Then there is the csv module which is used for getting the data from the excel files.
And finally, msvcrt module is used to allow the user to go back to the main menu by pressing any key.
Source code: https://github.com/NedkoNedkov001/HousingAgencyLinnaeusUniversity
Video demonstration: https://www.youtube.com/watch?v=Al7GSBWiBdk

Changelog

Person	Task	Date
Nedko	Setting-up server environment and Git repository	2022-03-10
Nedko	Designing of the database schema	2022-03-10
Tom	Creating of the data for populating the database	2022-03-10
Nedko	Creation of the database with python	2022-03-12
Tom	Populating the database	2022-03-13
Nedko	Creation of the queries and documentation in the report.	2022-03-14
Tom	Documentation in the report and creation of the video	2022-03-15