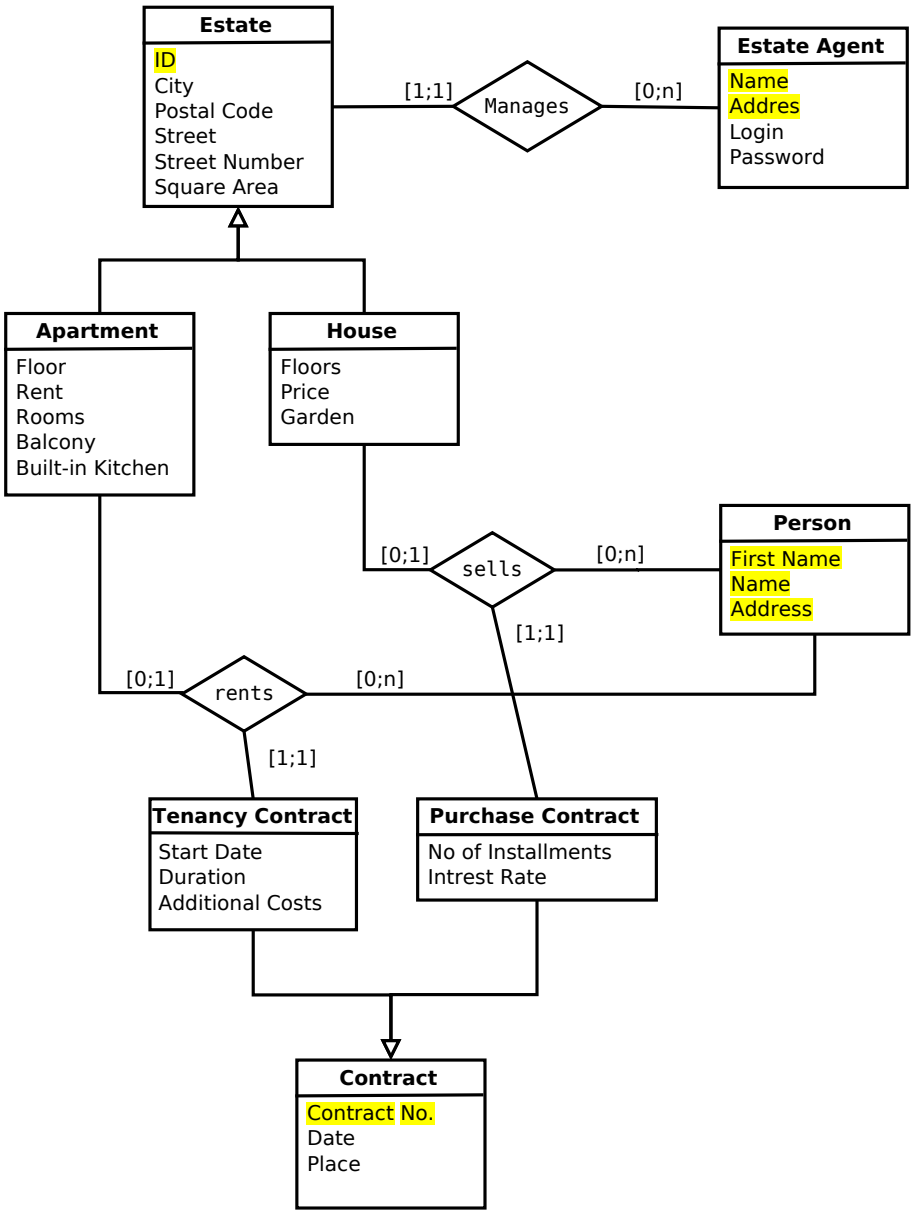
	Course	**Databases and Information Systems**		2025
	Exercise Sheet	**2**		
	Points	–		
	Release Date	**April 15th 2025**	Due Date	**April 30th 2025**

1 Development of a relational database application

In this assignment, you will develop a DB-backed Java application for the management of real estate. This is the domain model:



The central entity is an estate agent that manages estates. It has a unique login name as well as a password.

There are two basic types of estates: houses and apartments. Apartments are rented, whereas houses are sold. For each estate, some general information are stored: identification number, address (comprised of city, postal code, street and number) and square area. Additionally, each apartment has a floor number, a rent (price), a certain number of rooms, a flag indicating whether there is a balcony and a flag indicating whether there is a built-in kitchen. Houses, on the other hand, have a number of floors, a price, and a flag for whether a garden is included.

For every tenancy and every sale, respectively, there is a formal contract which has a unique contract number, a contract date, and a settlement place. Tenancy agreements (for apartments) have a start date, a tenancy duration and extra charges (utilities). A house can be paid by installments. The amount of installments and the interest rate are part of the sale contract.

For every contract, there is only one renting/buying person, but each person can rent/buy an arbitrary amount of properties.

1.1 Database

You can either use the database provided in sheet 1 or use a local installation. If you use a local installation, you do not depend on the VPN or internet at all while working on your exercise. Again, you can choose how you install it locally:

1.1.1 Manual installation

A documentation for a local installation can be found here: https://www.postgresql.org/download/ You may also find PostgreSQL in your package manager.

1.1.2 Docker

Alternatively you could use docker[1]: Run the following command after you have installed Docker:

```
docker run --name dis -p 127.0.0.1:5432:5432 -e POSTGRES_PASSWORD=postgres postgres
```

With **docker stop dis** and **docker start dis** you can stop and start the server again.

Explanation: This will create and start a Docker PostgreSQL container with the name dis and expose the default PostgreSQL port to your localhost, so you can use it, as you would normally use PostgreSQL. The last part defines the password for your Postgres user.

[1] https://www.docker.com/get-started

1.2 DB-schema

Translate the above model to a relational model by defining the respective DB schema. Fulfill the following requirements:

- The commands for creating the DB objects are contained in SQL scripts.
- Choose an inheritance model (e.g., horizontal partitioning), note your thoughts on why you chose one over the other. We recommend, for this task, not to use PostgreSQL's internal inheritance feature.
- Define a primary key for every relation (surrogate keys are ok). Define foreign keys.
- Initialize the tables with appropriate sample data. There should, for instance, be an estate agent account you can use to log in.
- Create the tables using a GUI or the CLI

The tables created in this part of the assignment are the basis for the next part.

1.3 Java application

Implement an estate management using the previously created DB schema. The realization of the UI (graphical or command-line interface) is up to you.

The application should support the following functionalities:

- Management mode for estate agents
  - Account creation
  - Changing and deleting accounts

  To access this mode, the user has to enter a separate password, which is hard-coded in the application for simplicity.
- Management mode for estates
  - Estate agents can log in with their individual accounts
  - Creating, deleting, and updating estates
- Contract management
  - Insert persons
  - Sign (create) contracts
  - Overview of all contracts

There is a sample project in Moodle. For your implementation, you can either use this project and import it into the IDE of your choice or create your own project. Make sure that the JDBC driver in lib/ is added to your classpath. You also have to change the *db.properties* to match your database.
If you create your own project, don't forget to download and add the driver. (JDBC Driver for Java, Psycopg lib for Python).