CAT 1

Database Design

Alejandro Pérez Bueno

Oct 21, 2024

Table of Contents

Exercise 1

Axil is a real estate network currently undergoing expansion, and they have commissioned us to design a database that efficiently handles information related to their real estate agencies distributed throughout the country.

Each real estate agency is assigned an alphanumeric code that identifies it. The database must also store the agency's address, phone number, email, and website, if available.

Each real estate agency is located in a city, which needs to be known. Each city is identified by its name, and we also want to store its area in square kilometers and its population. Cities without any Axil real estate agency can be stored in the database, and it's known that a city may have more than one real estate agency.

Each real estate agency keeps a record of its clients. Each client is assigned to only one agency. When a client is registered, they are given a numeric code that identifies them, and we want to record their NIF (tax identification number) or NIE (foreign identification number), their name, address, phone number, email (if provided), and the city where they reside. It's possible for cities to be recorded in the database where no clients reside.

Given that the volume of foreign clients is high, it is important to register the nationality or nationalities (maximum of two) of the clients for each agency. To this end, the database will store the following information about the countries whose nationalities Axil's clients hold: ISO code, which is a 2-character alphanumeric string identifying the country, the country's name, and the continent to which it belongs (Europe, Asia, Africa, America, or Oceania). It is known that the database may include information about countries whose nationality does not correspond to any client, and that each client will have one or two nationalities recorded.

Relationships and Cardinalities

- **Real Estate Agency to City**: Many-to-One
  - Each agency is located in one city, but a city can have multiple agencies.
- **Real Estate Agency to Client**: One-to-Many
  - Each client is assigned to one agency, but an agency can have multiple clients.
- **Client to City**: Many-to-One
  - Each client resides in one city, but a city can have multiple clients.
- **Client to Country**: Many-to-Many
  - Each client can have one or two nationalities, and each country can be associated with multiple clients.

UML Class Diagram
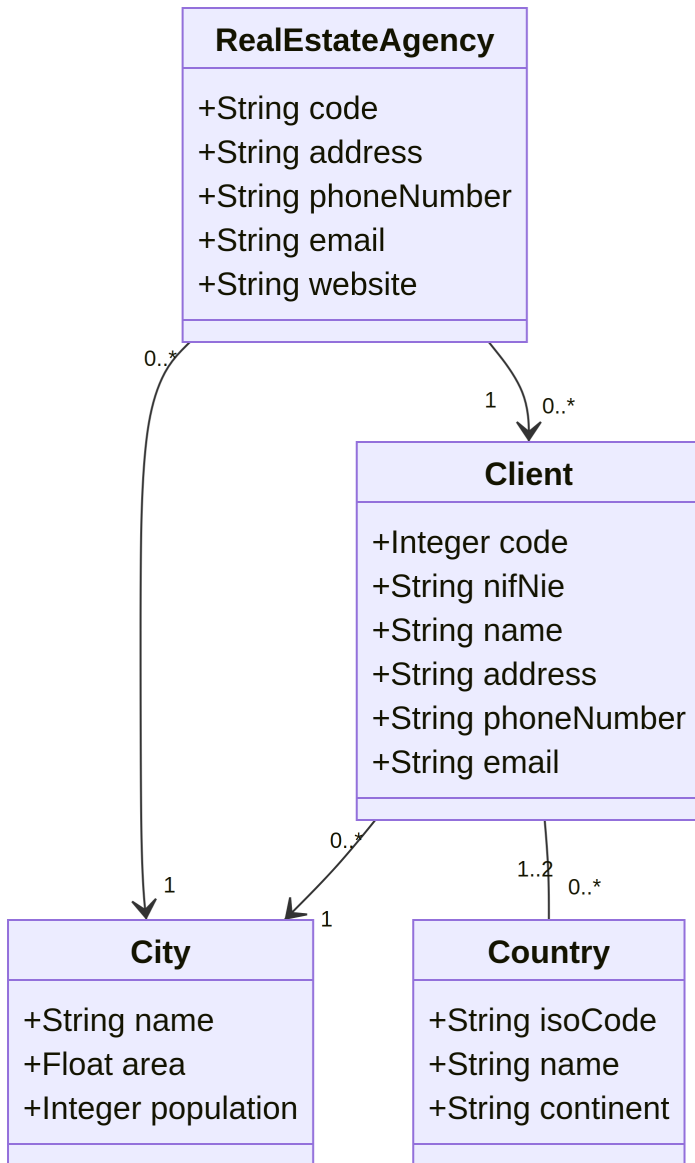
Additional Considerations

- **Primary Keys**:
  - **RealEstateAgency**: `code`
  - **City**: `name`
  - **Client**: `code`
  - **Country**: `isoCode`
- **Foreign Keys**:
  - `RealEstateAgency` has a foreign key to `City`.
  - `Client` has foreign keys to `RealEstateAgency` and `City`.
  - A separate associative entity or table may be needed to handle the many-to-many relationship between `Client` and `Country`.

Exercise 2

Axil also wants us to record information about its employees in the database.

Each employee is identified by their social security number (NSS), which is an alphanumeric code. The database should also store their first and last names, address, phone number, salary, and the city where they reside. The database may include cities where no employees reside.

There are only two types of employees at Axil: sales agents and administrative staff. It is not possible for an employee to perform both roles simultaneously.

Each sales agent is assigned to only one real estate agency, and it is required that at least one sales agent works at each agency. For each sales agent, we need to know whether they hold the position of agency manager and the commission percentage they receive (between 2% and 10%), which may be a decimal number.

On the other hand, administrative staff can be assigned to more than one real estate agency (at least one), in which case it is necessary to know the percentage of their workday dedicated to each of the agencies they work for, ensuring that the total percentage for an administrative employee working in multiple agencies adds up to 100%. For all administrative staff, the date they joined the company must also be recorded. It is known that more than one administrative employee can work at a real estate agency, but there are small agencies where no administrative staff are employed.

To address Exercise 2, we need to extend the model to include employees and their specific roles. Here's a breakdown of the entities and relationships:

Entities and Attributes

1. **Employee**
   - Attributes:
     - NSS (Primary Key)
     - First Name
     - Last Name
     - Address
     - Phone Number
     - Salary
     - Residing City
2. **SalesAgent** (inherits from Employee)
   - Attributes:
     - IsManager (Boolean)
     - CommissionPercentage (Decimal between 2% and 10%)
   - Relationships:
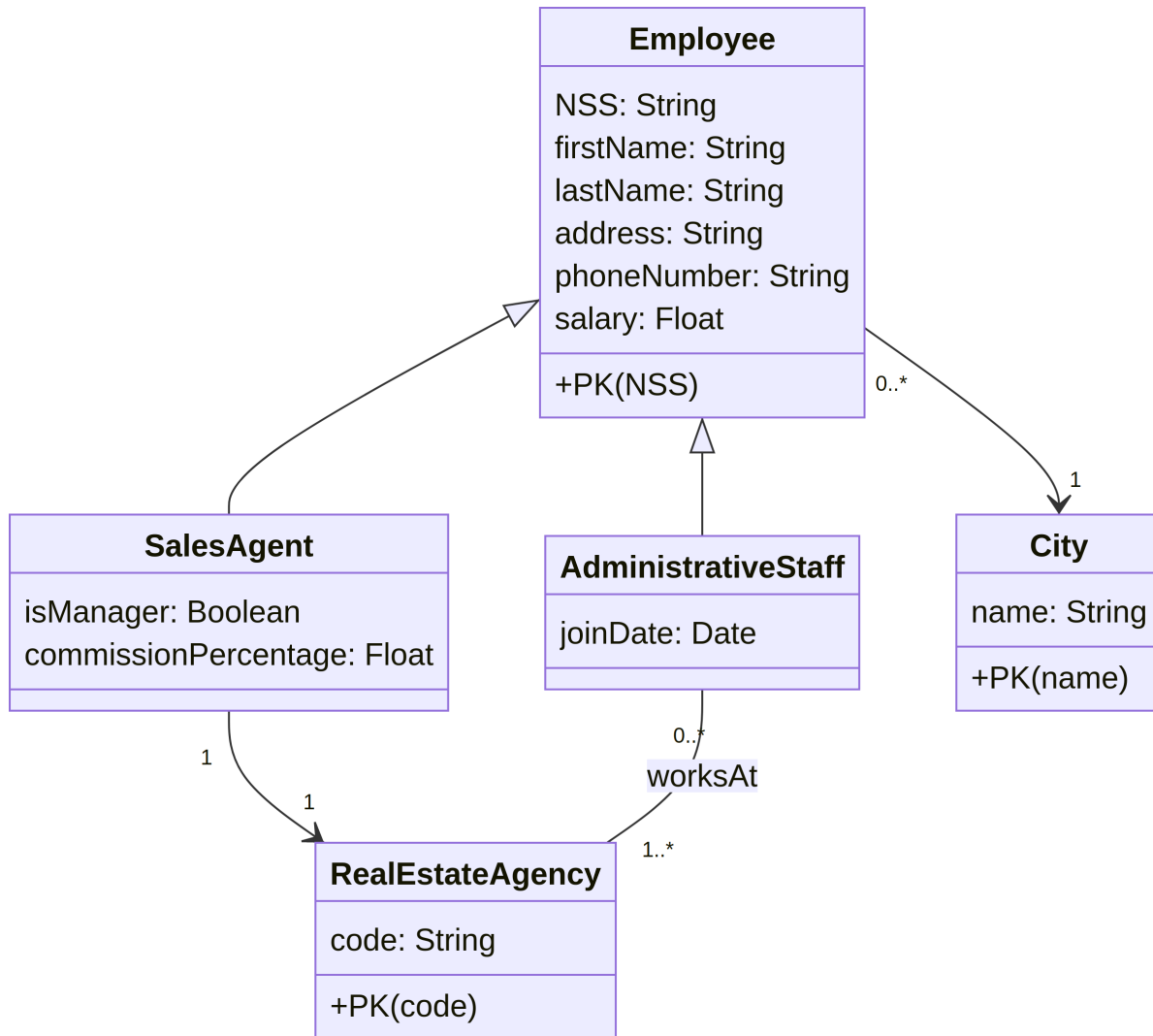   - Assigned to one Real Estate Agency
3. **AdministrativeStaff** (inherits from Employee)
   - Attributes:
     - JoinDate
   - Relationships:
     - Can be assigned to multiple Real Estate Agencies with a workday percentage

Relationships and Cardinalities

- **SalesAgent to RealEstateAgency**: One-to-One
  - Each sales agent is assigned to one agency, and each agency must have at least one sales agent.
- **AdministrativeStaff to RealEstateAgency**: Many-to-Many
  - Administrative staff can work for multiple agencies, with a workday percentage for each.

UML Class Diagram

Employee

NSS: String
firstName: String
lastName: String
address: String
phoneNumber: String
salary: Float

+PK(NSS)

0..*

1

SalesAgent

isManager: Boolean
commissionPercentage: Float

AdministrativeStaff

joinDate: Date

City

name: String

+PK(name)

0..*

worksAt

1

1

1..*

RealEstateAgency

code: String

+PK(code)

Key Points

- **Inheritance**: `SalesAgent` and `AdministrativeStaff` inherit from `Employee`.
- **Primary Keys**: Specified for each class.
- **Relationships**: Defined according to the requirements, with cardinalities reflecting the constraints.
- **Constraints**: Ensure that each `SalesAgent` is linked to one `RealEstateAgency`, and `AdministrativeStaff` can be linked to multiple agencies with workday percentages.

Exercise 3

Entities and Attributes

1. **Province**
   - Attributes:
     - code (Primary Key)
     - name (Unique)
2. **City**
   - Attributes:
     - name
     - area
     - population
   - Relationships:
     - Belongs to one Province
3. **RealEstateAgency**
   - Attributes:
     - code (Primary Key)
     - address
     - phoneNumber
     - email
     - website
   - Relationships:
   - Has multiple OpeningSchedules
4. **OpeningSchedule**
   - Attributes:
     - dayOfWeek
     - firstOpeningTime
     - firstClosingTime
     - secondOpeningTime
     - secondClosingTime
   - Relationships:
     - Associated with one RealEstateAgency

Relationships and Cardinalities

- **City to Province**: Many-to-One
  - Each city belongs to one province, but a province can have multiple cities.
- **RealEstateAgency to OpeningSchedule**: One-to-Many
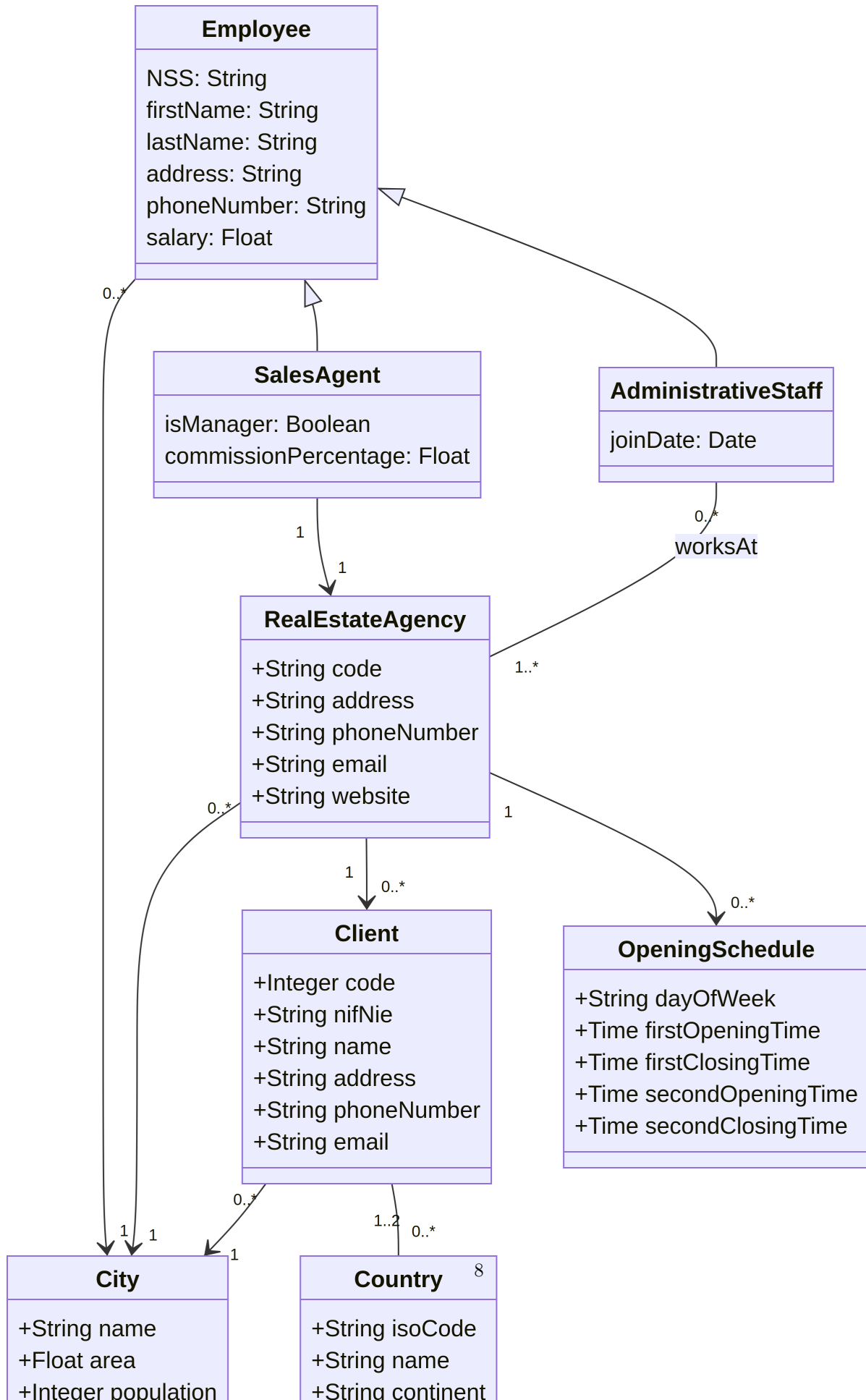 - Each agency can have multiple opening schedules, one for each day of the week.

UML Class Diagram
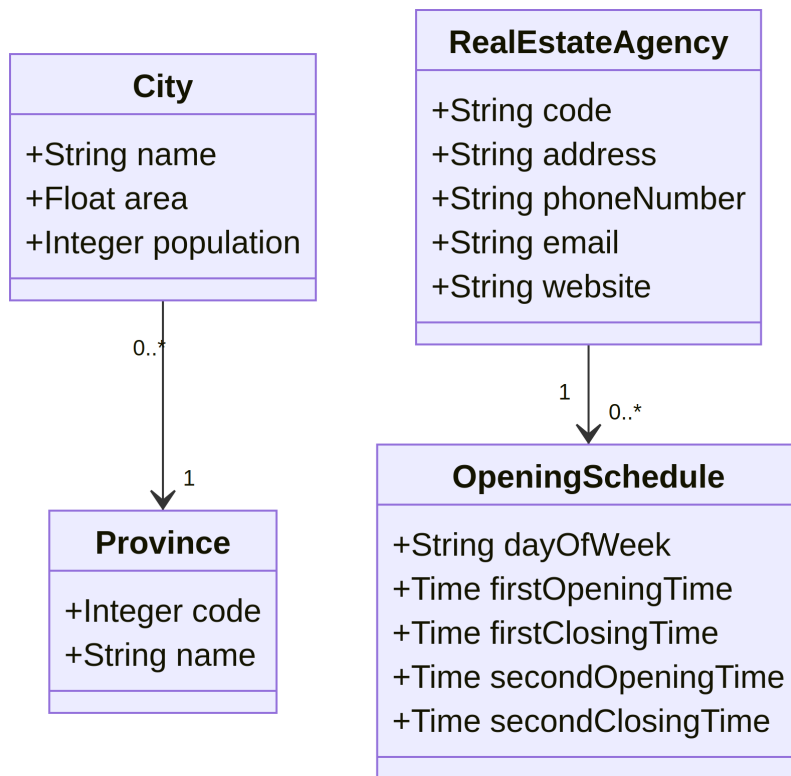
Key Points

- **Province**: Added to uniquely identify cities.
- **City**: Now includes a relationship to Province.
- **OpeningSchedule**: New entity to handle agency schedules.
- **Relationships**: Defined to reflect the new requirements.

Exercise 4

Key Points

- **Province**: Added to uniquely identify cities.
- **City**: Linked to Province and used by Clients and Employees.
- **RealEstateAgency**: Linked to City, Clients, and OpeningSchedules.
- **Client**: Linked to City and Country.
- **Employee**: Generalized with specific roles for SalesAgent and AdministrativeStaff.
- **OpeningSchedule**: Handles agency schedules.

Exercise 5

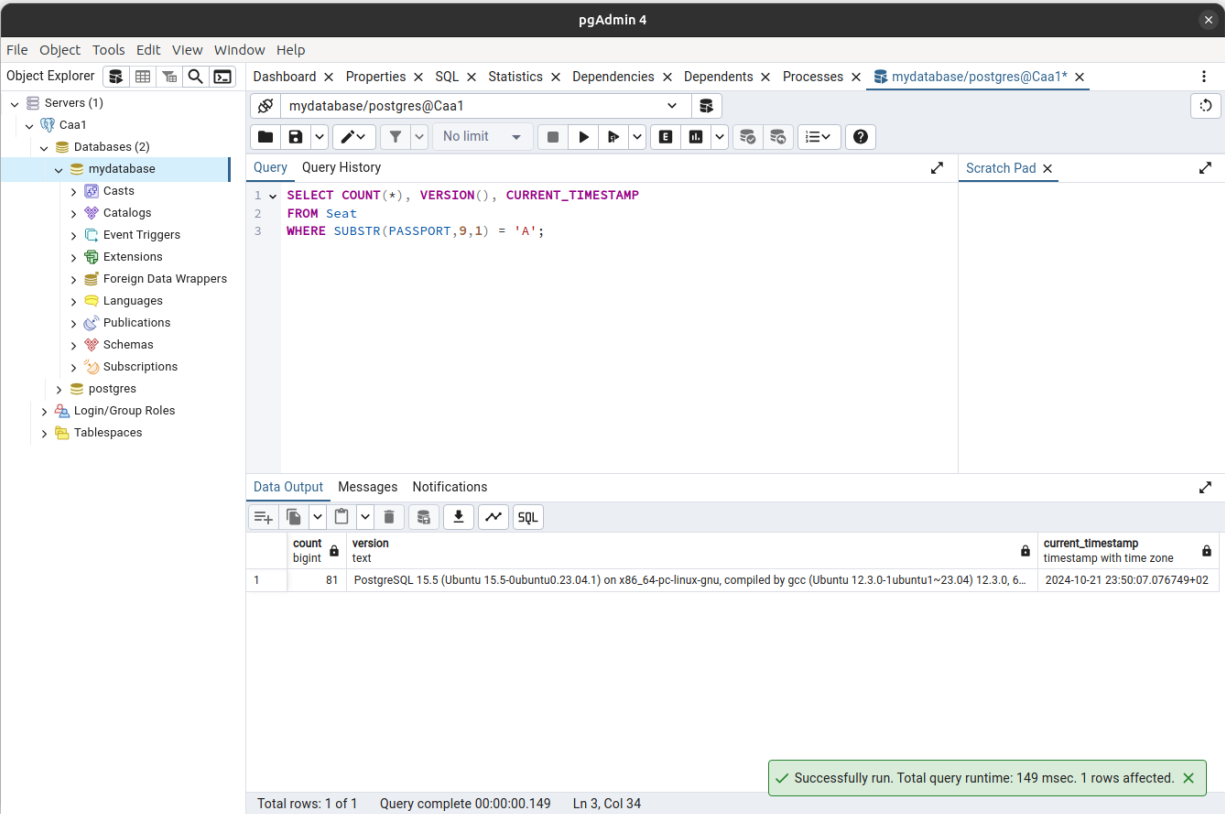Below is the screenshot after configuring the PostgreSQL Server and running the command that has executed the lines in *seat.txt*:

Figure 1: Seat query result