

Database Design for Privat Transport

University Exam

May 31, 2024

Introduction

This document details the database design process for Privat Transport, a private transport company established in Grimstad in 2023. The design follows a structured methodology comprising conceptual, logical, and physical database design phases. The objective is to create a database application to support the operations of Privat Transport, addressing issues of information sharing and administrative efficiency.

Conceptual Database Design

The conceptual database design involves constructing a model of the data requirements independent of physical considerations. The steps include identifying entity types, relationship types, attributes, and keys, and validating the model against user transactions.

Step 1.1: Identify Entity Types

| Entity Types |
|---------------------|
| Office |
| Manager |
| CarOwner |
| Driver |
| AdministrativeStaff |
| Car |
| PrivateClient |
| BusinessClient |
| Contract |
| Job |

Step 1.2: Identify Relationship Types

| Relationship Types |
|------------------------------------|
| Manager Manages Office |
| CarOwner Owns Car |
| Driver Drives Car |
| Office Employs AdministrativeStaff |
| PrivateClient Requests Job |
| BusinessClient Has Contract |
| Contract Includes Job |
| Job AssignedTo Driver |
| Job Uses Car |

Step 1.3: Identify and Associate Attributes

| Entity | Attributes |
|---------------------|--|
| Office | officeID, city |
| Manager | managerID, name, phone, officeID |
| CarOwner | ownerID, name, phone |
| Driver | driverID, name, sex, DOB, phone, officeID |
| AdministrativeStaff | staffID, name, phone, officeID |
| Car | carID, registrationNo, model, ownerID |
| PrivateClient | clientID, name, phone, address |
| BusinessClient | clientID, name, phone, address |
| Contract | contractID, clientID, numberOfJobs, fee |
| Job | jobID, clientID, contractID, pickUpDate-Time, pickUpAddress, dropOffAddress, mileage, charge, status |

Step 1.4: Determine Attribute Domains

| Attribute | Domain |
|--|----------------------------|
| officeID | Integer |
| managerID, ownerID, driverID, staffID, clientID, contractID, jobID | Integer |
| name, phone, address, model, registrationNo | String |
| sex | Char (M, F) |
| DOB, pickUpDateTime | Date |
| numberOfJobs, mileage | Integer |
| fee, charge | Decimal |
| status | String (Completed, Failed) |

Step 1.5: Determine Candidate, Primary, and Alternate Key Attributes

| Entity | Primary Key |
|---------------------|-------------|
| Office | officeID |
| Manager | managerID |
| CarOwner | ownerID |
| Driver | driverID |
| AdministrativeStaff | staffID |
| Car | carID |
| PrivateClient | clientID |
| BusinessClient | clientID |
| Contract | contractID |
| Job | jobID |

Step 1.6: Consider Use of Enhanced Modeling Concepts (Optional)

| Enhanced Modeling Concepts |
|---|
| Specialization: Client superclass with PrivateClient and BusinessClient subclasses. |

Step 1.7: Check Model for Redundancy

| Redundancy Check |
|--|
| Ensure no redundant entities or relationships exist. |

Step 1.8: Validate Conceptual Model Against User Transactions

| Validation |
|---|
| Validate that the model supports all user transactions such as listing managers, female drivers, total staff, car details, etc. |

Step 1.9: Review Conceptual Data Model with User

| Review |
|---|
| Review the ER diagram and associated documentation with the user. |

For this step we would simply review if the ER diagram below encompasses the requirements from the them.

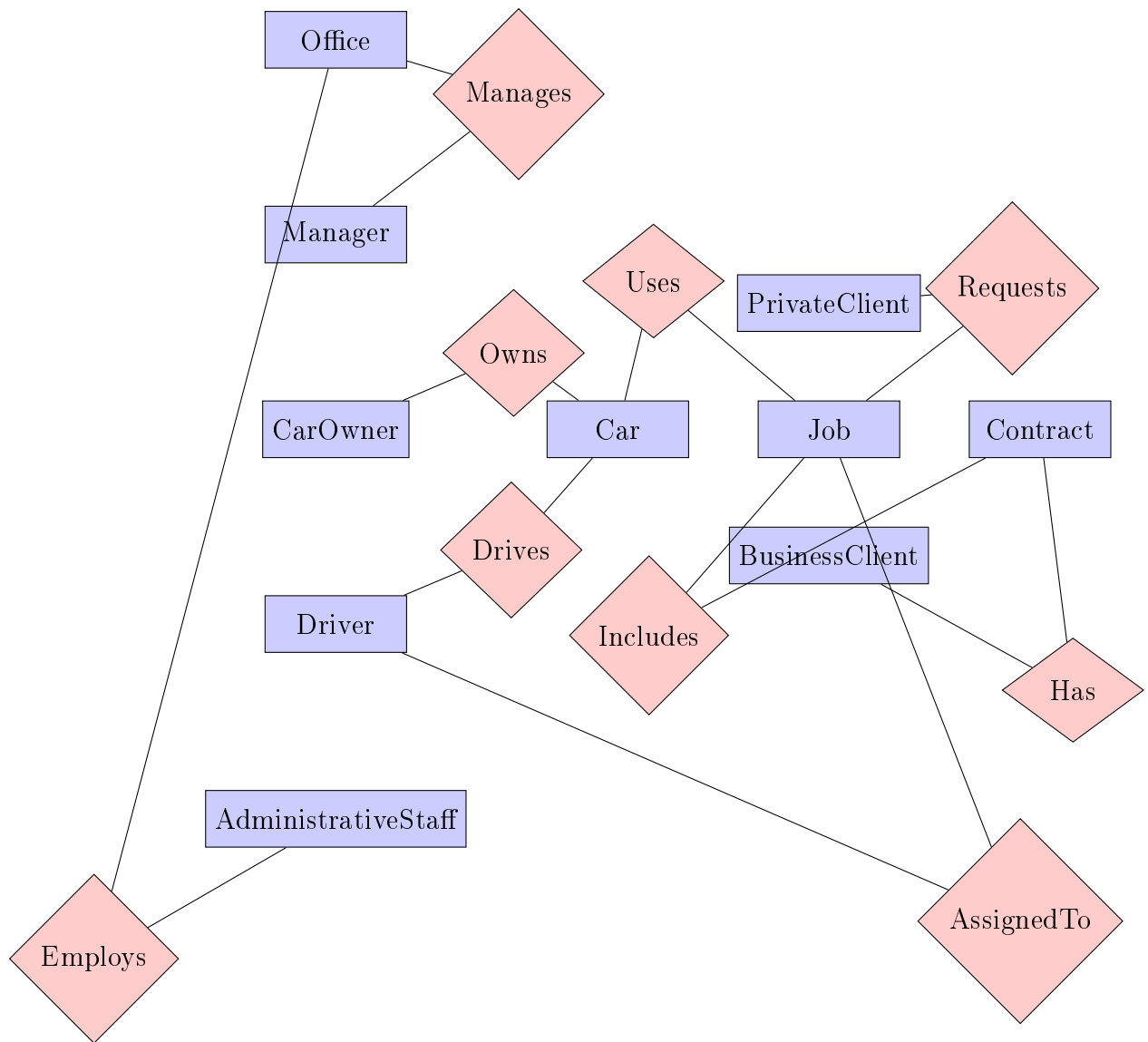


Figure 1: ER Diagram for Privat Transport

Logical Database Design

The logical database design translates the conceptual data model into a relational model. The steps include deriving relations, validating them using normalization, and ensuring they support user transactions.

Step 2.1: Derive Relations for Logical Data Model

| Relations for Logical Data Model |
|--|
| Office(officeID, city) |
| Manager(managerID, name, phone, officeID) |
| CarOwner(ownerID, name, phone) |
| Driver(driverID, name, sex, DOB, phone, officeID) |
| AdministrativeStaff(staffID, name, phone, officeID) |
| Car(carID, registrationNo, model, ownerID) |
| PrivateClient(clientID, name, phone, address) |
| BusinessClient(clientID, name, phone, address) |
| Contract(contractID, clientID, numberOfJobs, fee) |
| Job(jobID, clientID, contractID, pickUpDateTime, pickUpAddress, dropOffAddress, mileage, charge, status) |

Step 2.2: Validate Relations Using Normalization

| Normalization Validation |
|---|
| Ensure all relations are in at least Third Normal Form (3NF) to eliminate redundancy and ensure data integrity. |

Step 2.3: Validate Relations Against User Transactions

| User Transaction Validation |
|--|
| Check that each relation supports the necessary user transactions. |

Step 2.4: Define Integrity Constraints

| Integrity Constraints |
|---|
| Ensure all primary keys, foreign keys, and other constraints are correctly defined. |

Step 2.5: Review Logical Data Model with User

| Review |
|--|
| Review the logical data model with the user for accuracy and completeness. |

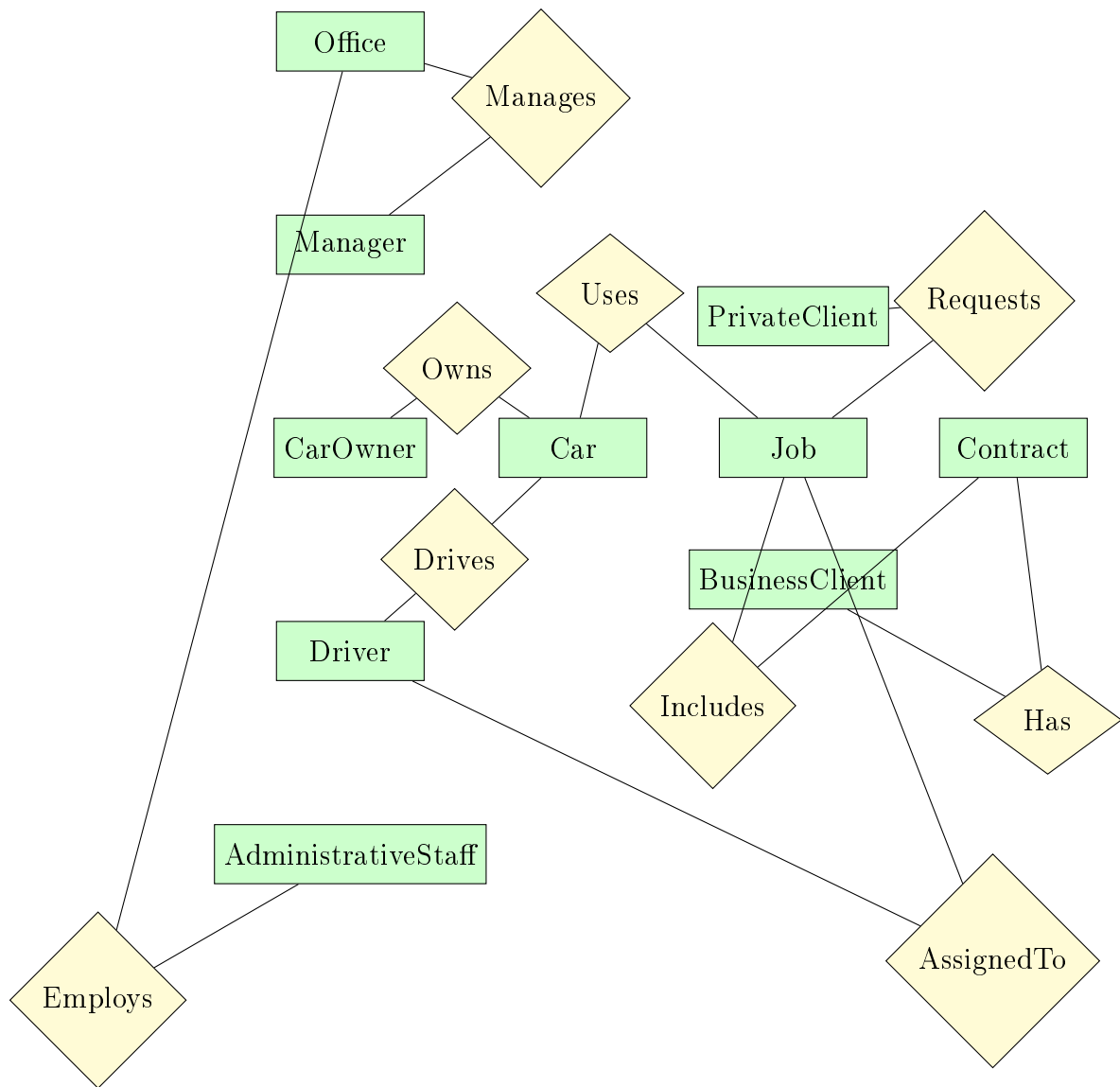


Figure 2: ER Diagram for Logical Data Model

Physical Database Design

The physical database design involves translating the logical data model into a physical implementation on MySQL. This includes defining the SQL schema and constraints.

Step 3.1: Translate Logical Data Model for MySQL

Create SQL scripts for each table based on the logical data model.

SQL Implementation

Create Tables

```
CREATE TABLE Office (
  officeID INT PRIMARY KEY,
  city VARCHAR(100)
```

```

);

CREATE TABLE Manager (
    managerID INT PRIMARY KEY,
    name VARCHAR(100),
    phone VARCHAR(15),
    officeID INT,
    FOREIGN KEY (officeID) REFERENCES Office(officeID)
);

CREATE TABLE CarOwner (
    ownerID INT PRIMARY KEY,
    name VARCHAR(100),
    phone VARCHAR(15)
);

CREATE TABLE Driver (
    driverID INT PRIMARY KEY,
    name VARCHAR(100),
    sex CHAR(1),
    DOB DATE,
    phone VARCHAR(15),
    officeID INT,
    FOREIGN KEY (officeID) REFERENCES Office(officeID)
);

CREATE TABLE AdministrativeStaff (
    staffID INT PRIMARY KEY,
    name VARCHAR(100),
    phone VARCHAR(15),
    officeID INT,
    FOREIGN KEY (officeID) REFERENCES Office(officeID)
);

CREATE TABLE Car (
    carID INT PRIMARY KEY,
    registrationNo VARCHAR(10),
    model VARCHAR(50),
    ownerID INT,
    FOREIGN KEY (ownerID) REFERENCES CarOwner(ownerID)
);

CREATE TABLE PrivateClient (
    clientID INT PRIMARY KEY,
    name VARCHAR(100),
    phone VARCHAR(15),
    address VARCHAR(255)
);

CREATE TABLE BusinessClient (
    clientID INT PRIMARY KEY,

```

```

        name VARCHAR(100),
        phone VARCHAR(15),
        address VARCHAR(255)
    );

CREATE TABLE Contract (
    contractID INT PRIMARY KEY,
    clientID INT,
    numberOfJobs INT,
    fee DECIMAL(10,2),
    FOREIGN KEY (clientID) REFERENCES BusinessClient(clientID)
);

CREATE TABLE Job (
    jobID INT PRIMARY KEY,
    clientID INT,
    contractID INT,
    pickUpDateTime DATETIME,
    pickUpAddress VARCHAR(255),
    dropOffAddress VARCHAR(255),
    mileage INT,
    charge DECIMAL(10,2),
    status VARCHAR(20),
    FOREIGN KEY (clientID) REFERENCES PrivateClient(clientID),
    FOREIGN KEY (contractID) REFERENCES Contract(contractID)
);

```

Conclusion

This report shows the complete database design process for Privat Transport, including conceptual, logical, and physical design phases. The models and SQL scripts provided ensure a robust database implementation suitable for the company's needs.

REFERENCES

The used references are class documents(pdf, powerpoints etc.) mostly on chapter 16.