Database Design and Modeling

University Exam

May 31, 2024

Introduction

This document provides a detailed methodology for database design and modeling, focusing primarily on the conceptual and logical design phases. The methodology follows the steps outlined in the provided Connolly and Begg's database design guidelines.

Step 1: Conceptual Database Design

The conceptual database design phase involves building a conceptual representation of the database that includes identifying the important entities, relationships, and attributes. The steps are as follows:

Step 1.1: Identify Entity Types

Identify the main objects of interest (entities) within the given problem statement. These are typically nouns or noun phrases in the specification.

- **Staff**
- **PropertyForRent**
- **PrivateOwner**
- **BusinessOwner**
- **Client**
- **Preference**
- **Lease**

Step 1.2: Identify Relationship Types

Determine the relationships between the identified entities. Relationships are usually verbs or verb phrases.

- **Staff Manages PropertyForRent**
- **PrivateOwner Owns PropertyForRent**
- **PropertyForRent AssociatedWith Lease**

Step 1.3: Identify and Associate Attributes with Entity or Relationship Types

List the attributes for each entity or relationship.

- **Staff**: staffNo, fName, lName, position, sex, DOB
- **PropertyForRent**: propertyNo, address, type, rooms, rent
- **PrivateOwner**: ownerNo, name, address, telNo
- **BusinessOwner**: ownerNo, bName, bType, address, telNo, contactName
- **Client**: clientNo, fName, lName, telNo, eMail
- **Preference**: prefType, maxRent
- **Lease**: leaseNo, paymentMethod, deposit, depositPaid, rentStart, rentFinish

Step 1.4: Determine Attribute Domains

Define the domain for each attribute.

- **staffNo**: String(5)
- **fName, lName**: String(50)
- **position**: String(30)
- **sex**: Char(1) 'M', 'F'
- **DOB**: Date
- **propertyNo**: String(5)
- **address**: Composite street: String(100), city: String(50), postcode: String(10)
- **type**: String(20)
- **rooms**: Integer
- **rent**: Decimal
- **ownerNo**: String(5)
- **name, bName, contactName**: String(50)
- **bType**: String(20)
- **telNo**: String(15)
- **clientNo**: String(5)
- **telNo, eMail**: String(50)
- **prefType**: String(20)

- **maxRent**: Decimal
- **leaseNo**: String(5)
- **paymentMethod**: String(20)
- **deposit, depositPaid**: Decimal
- **rentStart, rentFinish**: Date

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

Identify the candidate keys for each entity and choose the primary key.

- **Staff**: Primary Key = staffNo
- **PropertyForRent**: Primary Key = propertyNo
- **PrivateOwner**: Primary Key = ownerNo
- **BusinessOwner**: Primary Key = ownerNo
- **Client**: Primary Key = clientNo
- **Preference**: Composite Key = clientNo, prefType
- **Lease**: Primary Key = leaseNo

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

Use enhanced ER modeling concepts like generalization/specialization, aggregation, or composition if necessary. For instance, PrivateOwner and BusinessOwner can be specialized from Owner.

Step 1.7: Check Model for Redundancy

Re-examine the model to ensure there are no redundant entities or relationships.

Step 1.8: Validate Conceptual Data Model Against User Transactions

Ensure that the conceptual data model supports all the user transactions by mapping each transaction to the data model.

Step 1.9: Review Conceptual Data Model with User

Review the conceptual data model with the user to ensure it meets their requirements and expectations.

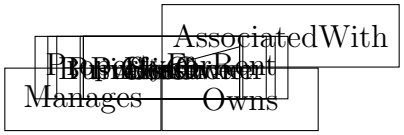

Figure 1: ER Diagram for Conceptual Data Model

Step 2: Logical Database Design

The logical database design phase translates the conceptual data model into a logical structure that can be implemented using a DBMS such as MySQL or SQLite.

Step 2.1: Derive Relations for Logical Data Model

Create relations for each entity and relationship identified in the conceptual model.

- **Staff (staffNo, fName, lName, position, sex, DOB)**
- **PropertyForRent (propertyNo, address, type, rooms, rent)**
- **PrivateOwner (ownerNo, name, address, telNo)**
- **BusinessOwner (ownerNo, bName, bType, address, telNo, contactName)**
- **Client (clientNo, fName, lName, telNo, eMail, staffNo)**
- **Preference (clientNo, prefType, maxRent)**
- **Lease (leaseNo, propertyNo, clientNo, paymentMethod, deposit, depositPaid, rentStart, rentFinish)**

Step 2.2: Validate Relations Using Normalization

Ensure that each relation is in at least the Third Normal Form (3NF) to avoid redundancy and ensure data integrity.

Step 2.3: Validate Relations Against User Transactions

Check that each relation supports the necessary user transactions. For example, listing properties managed by a specific staff member would involve joining the *Staff* and *PropertyForRent* tables.

Step 2.4: Check Integrity Constraints

Ensure all primary keys, foreign keys, and other constraints are correctly applied.

Step 2.5: Review Logical Data Model with User

Review the logical data model with the user to confirm that it accurately reflects their requirements.

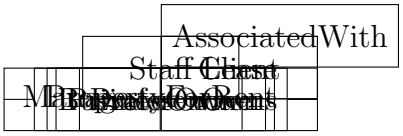

Figure 2: ER Diagram for Logical Data Model

Step 3: Physical Database Design (Not Covered)

This document focuses on the first two steps of the database design methodology. The physical design phase, which involves implementing the logical data model in a specific DBMS, is not covered.

SQL Implementation

To demonstrate the implementation, below are sample SQL statements for creating tables in MySQL/SQLite based on the logical model.

Create Tables

```
CREATE TABLE Staff (
    staffNo VARCHAR(5) PRIMARY KEY,
    fName VARCHAR(50),
    lName VARCHAR(50),
    position VARCHAR(30),
    sex CHAR(1),
    DOB DATE
);

CREATE TABLE PropertyForRent (
    propertyNo VARCHAR(5) PRIMARY KEY,
    address VARCHAR(100),
    type VARCHAR(20),
    rooms INTEGER,
    rent DECIMAL
);

CREATE TABLE PrivateOwner (
    ownerNo VARCHAR(5) PRIMARY KEY,
    name VARCHAR(50),
    address VARCHAR(100),
    telNo VARCHAR(15)
);

CREATE TABLE BusinessOwner (
    ownerNo VARCHAR(5) PRIMARY KEY,
    bName VARCHAR(50),
    bType VARCHAR(20),
```

```
    address VARCHAR(100),
    telNo VARCHAR(15),
    contactName VARCHAR(50)
);

CREATE TABLE Client (
    clientNo VARCHAR(5) PRIMARY KEY,
    fName VARCHAR(50),
    lName VARCHAR(50),
    telNo VARCHAR(50),
    eMail VARCHAR(50),
    staffNo VARCHAR(5),
    FOREIGN KEY (staffNo) REFERENCES Staff(staffNo)
);

CREATE TABLE Preference (
    clientNo VARCHAR(5),
    prefType VARCHAR(20),
    maxRent DECIMAL,
    PRIMARY KEY (clientNo, prefType)
);

CREATE TABLE Lease (
    leaseNo VARCHAR(5) PRIMARY KEY,
    propertyNo VARCHAR(5),
    clientNo VARCHAR(5),
    paymentMethod VARCHAR(20),
    deposit DECIMAL,
    depositPaid DECIMAL,
    rentStart DATE,
    rentFinish DATE,
    FOREIGN KEY (propertyNo) REFERENCES PropertyForRent(propertyNo),
    FOREIGN KEY (clientNo) REFERENCES Client(clientNo)
);
```