

Course ISE 322: Database Systems

Recitation 5 Exercise

Relational Model

Michael J. May

November 18, 2009

In this recitation we will do two tasks which center around understanding the relational model and how it works. First you will write pseudocode for enforcing primary key and foreign key constraints. Second, you will write code for creating tables based on the Kvissh 6 example with which we began the semester.

1 Enforcing Constraints

In class we discussed the importance of primary key constraints and foreign key constraints. To better understand their properties and the options attached to foreign key definitions (NO ACTION, CASCADE, SET DEFAULT, SET NULL) let us write pseudocode and queries to help the database enforce them.

1.1 Primary Key

To enforce primary key constraints we must first define precisely what we mean by a primary key constraint. For a table in which we define a single column as a primary key, we can describe the behavior of the database in the following pseudocode functions. In the following example, let the table be called “this” and the columns be stored in an array “this.columns”. The primary key is column pk which is part of this.columns. Write the pseudocode for the following functions: AddRow(newRow), DeleteRow(oldRow), UpdateRow(oldRow, newRow).

1.2 Foreign Key

To enforce foreign key constraints, we need a bit more logic in the add, delete, and update functions to take care of the different situations which may arise in the pointing/target relations. Assume that each foreign key is a relationship between one table and another and has two properties - one called “onDelete” and the other called “onUpdate”. You may assume that the foreign keys defined in a given table are either stored at the table level or at the column level. Modify the AddRow, DeleteRow, and UpdateRow functions you wrote above to reflect the logic required to enforce the foreign key constraints.

2 Creating Tables

Let us return for a moment to the Kvissh 6 example with which we began the semester. The tables to create are as follows:

1. Exits (exitId integer, milepost integer, name string)
2. AccountTypes (accountType string)

3. Segments (segmentId integer, direction string, startingExit integer, endingExit integer)
4. Customers (customerId integer, plateNumber integer, name string, balance real)
5. CustomerPaymentInfo (customerId integer, accountType string, accountNumber integer)
6. Trips (tripid integer, customerId integer, startExit integer, endExit integer)

Be sure to include all primary key and foreign key constraints.

3 Inserting Data

Write SQL insert statements to insert rows into all of the tables you have written. You may use the provided Excel sheets to help you write your SQL code for insertion. Pay attention to the order in which you must enter rows into the tables.

3.1 Queries

After you have prepared the above tables, write SQL to perform the following queries:

1. List the names and addresses of all customers.
2. List all customerIds which (1) do not have a name or (2) do not have a license plate number.
3. List of all customers with outstanding balances greater than 0.
4. List the name, address, account type, and account numbers of all customers who have payment information stored.
5. For each trip, print the name of the customer, the account type, and the names of the starting and ending exits.