**Homework Assignment 2**

Due: 11:59PM Saturday, May 2, 2020

|  |
| --- |
| * This is an individual work; Please be clear with HGU CSEE Standard:   + Submitting assignments or program codes written by others or acquired from the internet without explicit approval of the professor is regarded as cheating.   + Showing or lending one’s own homework to other student is also considered cheating that disturbs fair evaluation and hinders the academic achievement of the other student.   + It is regarded as cheating if two or more students conduct their homework together and submit it individually when the homework is not a group assignment. * Read the assignment carefully. You are **allowed to re-use any of the queries from the lecture slides** while developing solutions to the problems. * When finished, submit your work to *LMS.* |

**1. (1 pt. per each blank – total 17 blanks) Read textbook sections 2.3-2.5, 3-3.6 and 4-4.4, and fill in the blanks.**

(a) ( ) is the principal language used to describe and manipulate relational databases. It makes a distinction between three kinds of relations: ( ) relations, ( ), and ( ) tables.

(b) A ( ) asserts that a value appearing in one table also appears in another related table.

(c) We say one or more attributes is a ( ) for a relation if the attributes ( ) all other attributes of the relation, while no subset of the attributes does so.

(d) Sometimes, a relation has more than one key, where one of them is designated as the ( ).

(e) Redundancies in database design may occur different kinds of anomalies. ( ) refer to the case when a set of values becomes empty, one may lose other information as a side effect. ( ) refer to the case when one change information in one tuple but leave the same information in another tuple unchanged.

(f) An ( ) is a graph representing entity sets, attributes, and relationships. It is a notation for describing ( ) of databases.

(g) An ( ) is an abstract object of some sort, and a collection of similar entities forms an ( ).

(h) ( ) are connections among two or more entity sets.

(i) A ( ) for an entity set is an attribute or a set of attributes such that, given any two distinct entities, both cannot have identical values for each of the attributes in it.

(j) For ( ) entity sets, the key belongs to another entity set.

**2. Draw the E-R diagrams for the following databases. Be sure to indicate the cardinalities to indicate the multiplicity of a relationship.**

(a) Design a database for a bank, including information about customers and their accounts. Information about a customer includes their name, address, phone, and customer ID. An Account has an account number, balance, and opening date. Also record the customer(s) who own an account. Note that:  
- A customer can have multiple accounts.

- An account can have only one customer.

- Customer ID and Account number are unique to each customer and account, respectively.

(b) Modify your original diagram of Problem 2(a) such that a customer can have multiple phone numbers. Recall that we do not allow attributes to have non-primitive data types.

(c) Change your diagram of Problem 2(b) such that a customer has an address represented by composite attributes (which are street-city-province triplets). Note that multiple customers may live at a single address.

(d) Convert your solution E-R diagram from Problem 2(c) into database schemas (*e.g.*, TableName(Attribute1, Attributes2, Attribute3)).

(e) Convert your solution to Problem 2(d) into SQL DDL (CREATE TABLE statements).

3. Normalization.

(a) Is every relation in 3NF also in BCNF? If yes, explain why. If no, given a counter example.

(b) Is every relation in 4NF also in BCNF? If yes, explain why. If no, given a counter example.

(c) Consider the following relation. Does the relation violate any of the normal forms {1NF, 2NF, 3NF, 4NF, BCNF}? If yes, indicate which of the NFs is violated, explain why it is violated and how it could be fixed.

R1(employee\_id, name, branch, project)

|  |  |  |  |
| --- | --- | --- | --- |
| **employee\_id** | **name** | **branch** | **project** |
| 1001 | Brown | Seoul | Mars, Saturn |
| 1004 | Green | Seoul | Saturn |
| 1005 | Hopkins | Pohang | Jupiter, Saturn |
| 2001 | Kim | Seoul | Mars |
| 3002 | Walker | Seoul | Venus |
| 3004 | Kim | Pohang | Venus |
| 3005 | Clermont | Pohang | Jupiter, Saturn |

(d) Consider the following relation. Does the relation violate any of the normal forms {1NF, 2NF, 3NF, 4NF, BCNF}? If yes, indicate which of the NFs is violated, explain why it is violated and how it could be fixed.

R2(employee\_id, name, position, previous\_branch)

|  |  |  |  |
| --- | --- | --- | --- |
| **employee\_id** | **name** | **position** | **previous\_branch** |
| 1001 | Brown | Sales representative | Pohang |
| 1001 | Brown | Sales representative | Busan |
| 1005 | Hopkins | Software engineer | Seoul |
| 2001 | Kim | Software engineer | Busan |
| 3004 | Kim | Product manager | Seoul |
| 3004 | Kim | Product manager | Wonju |
| 3005 | Clermont | Project administrator | Seoul |

(e) Consider the following relation. Does the relation violate any of the normal forms {1NF, 2NF, 3NF, 4NF, BCNF}? If yes, indicate which of the NFs is violated, explain why it is violated and how it could be fixed.

R3(employee\_id, name, branch, branch\_address)

|  |  |  |  |
| --- | --- | --- | --- |
| **employee\_id** | **name** | **branch** | **branch\_address** |
| 1001 | Brown | Seoul | Garosu-gil 233 |
| 1004 | Green | Seoul | Garosu-gil 233 |
| 1005 | Hopkins | Pohang | Handong-ro 501 |
| 2001 | Kim | Seoul | Garosu-gil 233 |
| 3002 | Walker | Seoul | Garosu-gil 233 |
| 3004 | Kim | Pohang | Handong-ro 501 |
| 3005 | Clermont | Pohang | Handong-ro 501 |

**4. In this problem we use the "*sakila*" (DVD rental) database, which consists of 16 tables regarding movie inventory, actors, customers, rental history, payment information, *etc*. For each of the following questions, find answer based on the information recorded in the database and write a query that shows how you obtained the answer.**

(a) How many *stores* are found in the database?

Query:

Answer:

(b) How many unique *last names* are found in the *actor* relation?

Query:

Answer:

(c) According to the database, how many *inventories* (DVDs) have not been returned (the inventories that have not been returned do not have *return\_date*)?

Query:

Answer:

(d) How many distinct *customer*s have rented a movie title(s) from *staff\_id*=1?

Query:

Answer:

(e) How many distinct *film*s rated 'PG' are available?

Query:

Answer:

(f) List *three* most frequent *categories* of film available at *store\_id*=2 (if a store has multiple copies of the same film, consider each copy as an individual inventory).

Query:

Answer:

(g) How many *active* *customer*s are living in the *district* of 'England'?

Query:

Answer:

(h) Considering the rental history (*rental*) and payment history (*payment*), who has paid the largest amount of money for renting movies? List the *first* and *last name* of the *customer*, the total number of movie rentals, and total amount of money s/he has paid.

Query:

Answer:

(i) What is the *title* of the movie that has the longest *description* (*film\_text.description*) among the rental store with *store\_id*=2 has?

Query:

Answer:

(j) Which of the *film*s starred by 'FRED COSTNER' rented the most? Write the *title* of the film.

Query:

Answer:

(k) Using the '*customer\_list*' view, list all *name*s of people whose *address* is in the *city* of 'London'.

Query:

Answer:

(l) Write a query that uses only tables (does not use any views) and returns the same information as in Problem 4(k).

Query: