CSC540 (Fall 2009) Assignment 1 REFERENCE SOLUTIONS

Due date 9/10/2009 (at the beginning of class)

Grading scheme. Attach a sheet to the solutions explaining for each question, points deducted and why.

The main goal of grading is to allow think analytically about other solutions to the problems you worked on. You should try to identify errors in a given solution. There are two types of errors, minor and major. Minor errors typically result from careless mistakes whereas major errors might indicate a gap in understanding. For some of the questions, I will give examples of minor and major mistakes. For minor mistakes you can deduct 1/8th of the total points for the question, or ¼ if there are several minor mistakes. For major mistakes, you can deduct ¼ (if only 1) or ½ (if there are several of them). **There are a total of 97pts that are specified. The other 3pts (to make a total of 100) you give at your discretion depending on overall performance, clarity of presentation and so on.**

Qu 1. Total points (15pts)(a – 3pts, b – 3pts, c – 3pts, d – 6pts)

Goals. Correct identification of ER concepts e.g. a weak entity type and correction translation technique from ER model into Relational schema. The latter will test the rules for translations particularly for things like total participation.

For a, b and c the answers are either correct or not. For d, minor errors could be missing an attribute in a relation. Major errors (not identifying key, putting attribute in wrong relation)

Qu 2. Total points (20pts) There is not a single correct answer. You should ensure that the design captures all the constraints in the description. Minor errors are missing attributes in an entity type. Major errors are missing or incorrect representation of a constraint description in the model.

Qu. 3 . Total points (18pts). This should be graded independently of Qu. 2. In other words, even if qu 2 was wrong, you don't penalize the student here a second time. You only check if for whatever ER model they have in Qu.2 they have correctly translated into a relational schema described correctly in SQL. Minor mistakes are missing attributes, major mistakes are missing constraint specification e.g. Foreign key clause, or placing the wrong attribute in the wrong relation due to wrong merging of entity and relationship.

Qu. 4 Total points (12 – 3 pts each). Do not be concerned with format of update statement but more whether the description captures what is given in the answer. For the checks, just ensure that the checks indicated are the correct ones and that all the appropriate checks are mentioned.

Qu5. Total points (20 – 4pts each). You should check carefully the correctness of query (the answer is correct), the appropriateness of syntax (correct operator and conditions are specified, compatibility of operands to operators, etc).

Qu. 6 Total points (12pts – 2pts each) This is straightforward. If the set of tuples shown is correct, it is correct. Minor mistakes include wrong entry in one cell of table or something. Major errors, are missing joined tuples, or wrong tuples joined, etc).

Solutions

Part A Database Design

1. Solution:

   (a) Entity types: BANK, ACCOUNT, CUSTOMER, LOAN

   (b) Weak entity type: BANK-BRANCH. Partial key: BranchNo.

   Identifying relationship: BRANCHES.

   (c) The partial key BranchNo in BANK-BRANCH specifies that the same BranchNo value

   may occur under different BANKs. The identifying relationship BRANCHES specifies that

   BranchNo values are uniquely assigned for those BANK-BRANCH entities that are related

   to the same BANK entity. Hence, the combination of BANK Code and BranchNo together

   constitute a full identifier for a BANK-BRANCH.

   (d)

BANK(Code, Name, Addr)

ACCOUNT(Acct-no, Type, Balance, Branch-no, Code)

LOAN(Loan-no, Type, Amount, Branch-no, Code)

CUSTOMER(Ssn, Phone,Name, Addr)

A_C(Ssn, Acct-no)

L_C(Ssn, Loan-no)

BANK_BRANCH(Code, Branch-no, Addr)

A less efficient but correct translation is given below.

BANK(Code, Name, Addr)

ACCOUNT(Acct-no, Type, Balance)

LOAN(Loan-no, Type, Amount)

CUSTOMER(Ssn, Phone,Name, Addr)

A_C(Ssn, Acct-no)

L_C(Ssn, Loan-no)

BANK_BRANCH(Code, Branch-no, Addr)

BRANCHES(Code, Branch-no, Addr)

ACCTS(Acct-no, Branch-no, Code)

LOANS(Loan-no, Branch-no, Code)

2. Although you always wanted to be an artist, you ended up being an expert on databases because you love to cook data and you somehow confused *database* with *data baste*. Your old love is still there, however, so you set up a database company, ArtBase, that builds a product for art galleries. The core of this product is a database with a schema that captures all the information that galleries need to maintain. Galleries keep information about artists, their names (which are unique), birthplaces, age, and style of art. For each piece of artwork, the artist, the year it was made, its unique title, its type of art (e.g., painting, lithograph, sculpture, photograph), and its price must be stored. Pieces of artwork are also classified into groups of various kinds, for example, portraits, still lifes, works by Picasso, or works of the 19th century; a given piece may belong to more than one group. Each group is identified by a name (like those just given) that describes the group. Finally, galleries keep information about customers. For each customer, galleries keep that person’ s unique name, address, total amount of dollars spent in the gallery (very important!), and the artists and groups of art that the customer tends to like.

   Solution:

3. Write SQL statements to create the corresponding relations to the ER diagram you designed for Exercise 2. If your translation cannot capture any constraints in the ER diagram, explain why.

Solution:

The statements to create tables corresponding to entity sets Customer, Group, and Artist are straightforward and omitted. The other required tables can be created as follows:

```
1). CREATE TABLE Classify ( title CHAR(20),

name CHAR(20),

PRIMARY KEY (title, name),

FOREIGN KEY (title) REFERENCES Artwork Paints,

FOREIGN KEY (name) REFERENCES Group )


2). CREATE TABLE Like Group (name CHAR(20),

cust name CHAR(20),

PRIMARY KEY (name, cust name),

FOREIGN KEY (name) REFERENCES Group,

FOREIGN KEY (cust name) REFERENCES Customer)



3). CREATE TABLE Like Artist (name CHAR(20),

cust name CHAR(20),

PRIMARY KEY (name, cust name),

FOREIGN KEY (name) REFERENCES Artist,
```

FOREIGN KEY (cust name) REFERENCES Customer)

4). CREATE TABLE Artwork Paints (title CHAR(20),

artist name CHAR(20),

type CHAR(20),

price INTEGER,

year INTEGER,

PRIMARY KEY (title),

FOREIGN KEY (artist name)

References Artist)

PART B Relational Model and Algebra

4. Solution:

(a) One possible set of operations for the following update is the following:

INSERT <FNO,LNO,DT,SEAT_NO,CUST_NAME,CUST_PHONE> into SEAT_RESERVATION; MODIFY the LEG_INSTANCE tuple with the condition:

(FLIGHT_NUMBER=FNO AND LEG_NUMBER=LNO AND DATE=DT) by setting

NUMBER_OF_AVAILABLE_SEATS = NUMBER_OF_AVAILABLE_SEATS - 1; These operations should be repeated for each LEG of the flight on which a reservation is

made. This assumes that the reservation has only one seat. More complex operations will be needed for a more realistic reservation that may reserve several seats at once.

(b) We would check that NUMBER_OF_AVAILABLE_SEATS on each LEG_INSTANCE of the flight is greater than 1 before doing any reservation (unless overbooking is permitted),

and that the SEAT_NUMBER being reserved in SEAT_RESERVATION is available.

(c) The INSERT operation into SEAT_RESERVATION will check all the key, entity integrity,

and referential integrity constraints for the relation. The check that NUMBER_OF_AVAILABLE_SEATS on each LEG_INSTANCE of the flight is greater than 1

does not fall into any of the above types of constraints (it is a general semantic integrity

constraint).

(d) We will write a referential integrity constraint as R.A --> S (or R.(X) --> T)

whenever attribute A (or the set of attributes X) of relation R form a foreign key that

references the primary key of relation S (or T). FLIGHT_LEG.FLIGHT_NUMBER --> FLIGHT

FLIGHT_LEG.DEPARTURE_AIRPORT_CODE --> AIRPORT
FLIGHT_LEG.ARRIVAL_AIRPORT_CODE --> AIRPORT

LEG_INSTANCE.(FLIGHT_NUMBER,LEG_NUMBER) --> FLIGHT_LEG

LEG_INSTANCE.DEPARTURE_AIRPORT_CODE --> AIRPORT

LEG_INSTANCE.ARRIVAL_AIRPORT_CODE --> AIRPORT

LEG_INSTANCE.AIRPLANE_ID --> AIRPLANE

FARES.FLIGHT_NUMBER --> FLIGHT

CAN_LAND.AIRPLANE_TYPE_NAME --> AIRPLANE_TYPE

CAN_LAND.AIRPORT_CODE --> AIRPORT

AIRPLANE.AIRPLANE_TYPE --> AIRPLANE_TYPE

SEAT_RESERVATION (FLIGHT_NUMBER,LEG_NUMBER,DATE) --> LEG_INSTANCE

5. Solution:

(a) ρ(A, (BOOKCOPIESLIBRARY- BRANCH BOOK))

ρ (RESULT, (No_Of_Copies (BranchName='Sharpstown' Λ Title='The Lost Tribe'(A)))

Note: A better query would be to do the SELECTs before the JOIN as follows:

No_Of_Copies ((BranchName='Sharpstown' (LIBRARY-BRANCH)) (BOOKCOPIES (Title='The Lost Tribe'(BOOK))))

(b) BranchID,No_Of_Copies ((Title='The Lost Tribe' (BOOK)) BOOKCOPIES)

(c) ρ(NO_CHECKOUT_B, (CardNo (BORROWER) CardNo (BOOK_LOANS))

Name (BORROWER NO_CHECKOUT_B)

(d) ρ (S ,(BranchId (BranchName='Sharpstown' (LIBRARY-BRANCH)))

ρ (B_FROM_S ,(BookId,CardNo ((DueDate='today' (BOOKLOANS)) S))

Title,Name,Address (BOOK BORROWER B_FROM_S)

(g) ρ (SK(BookId,Title) , (AuthorName='Stephen King' (BOOK_AUTHORS)) BOOK)

ρ (CENTRAL(BranchId) , BranchName='Central' (LIBRARY_BRANCH))

Title,NoOfCopies (SK BOOKCOPIES CENTRAL)

6. Solution:

(a)

P	Q	R	A	B	C
10	a	5	10	b	6
10	a	5	10	b	5
25	a	6	25	c	3

(b)

P	Q	R	A	B	C

15	b	8	10	b	6
15	b	8	10	b	5

(c)

P	Q	R	A	B	C
10	a	5	10	b	6
10	a	5	10	b	5
15	b	8	null	null	null
25	a	6	25	c	3

(d)

P	Q	R	A	B	C
15	b	8	10	b	6
null	null	null	25	c	3
15	b	8	10	b	5

(e)

P	Q	R
10	a	5
15	b	8
25	a	6

10	b	6
25	c	3
10	b	5

(f)

P	Q	R	A	B	C
10	a	5	10	b	5