Homework 4A

5.11

Suppose that each of the following Update operations is applied directly to the database state shown in Figure 5.6. Discuss all integrity constraints violated by each operation, if any, and the different ways of enforcing these constraints.

1. Insert <’Robert’,’F’,’Scott’,’943775543’,’1972-06-21’,’2365 Newcastle Rd, Bellaire. TX’, M, 58000, ‘888665555’,1> into EMPLOYEE.

Ans= No constraint violations.

1. Insert into <’ProductA’, 4, ‘Bellaire’, 2> PROJECT.

Ans= Violates referential integrity because DNUM=2 and there is no tuple in the

DEPARTMENT relation with DNUMBER=2. We may enforce the constraint by: (i) rejecting

the insertion of the new PROJECT tuple, (ii) changing the value of DNUM in the new

PROJECT tuple to an existing DNUMBER value in the DEPARTMENT relation, or (iii)

inserting a new DEPARTMENT tuple with DNUMBER=2

1. Insert into<‘Production’, 4, ‘943775543’, ‘2007-10-01> DEPARTMENT.

Ans= Violates both the key constraint and referential integrity. Violates the key constraint

because there already exists a DEPARTMENT tuple with DNUMBER=4. We may enforce

this constraint by: (i) rejecting the insertion, or (ii) changing the value of DNUMBER

in the new DEPARTMENT tuple to a value that does not violate the key constraint. Violates

referential integrity because MGRSSN='943775543' and there is no tuple in the

EMPLOYEE relation with SSN='943775543'. We may enforce the constraint by: (i)

rejecting the insertion, (ii) changing the value of MGRSSN to an existing SSN value in

EMPLOYEE, or (iii) inserting a new EMPLOYEE tuple with SSN='943775543'.

1. Insert into<‘677678989’, NULL, ‘40.0’>WORKS\_ON.

Ans=Violates both the entity integrity and referential integrity. Violates entity integrity

because PNO, which is part of the primary key of WORKS\_ON, is null. We may enforce

this constraint by: (i) rejecting the insertion, or (ii) changing the value of PNO in the

new WORKS\_ON tuple to a value of PNUMBER that exists in the PROJECT relation.

Violates referential integrity because ESSN='677678989' and there is no tuple in the

EMPLOYEE relation with SSN='677678989'. We may enforce the constraint by: (i)

rejecting the insertion, (ii) changing the value of ESSN to an existing SSN value in

EMPLOYEE, or (iii) inserting a new EMPLOYEE tuple with SSN='677678989'

1. Insert into <‘453453453’, ‘John’, ‘M’, ‘1990-12-12’, ‘spouse’>DEPENDENT.

Ans=No constraint violations.

1. Delete the WORKS\_ON tuples with Essn = ‘333445555’.

Ans=No constraint violations.

1. Delete the EMPLOYEE tuple with Ssn = ‘987654321’.

Ans= Violates referential integrity because several tuples exist in the WORKS\_ON,

DEPENDENT, DEPARTMENT, and EMPLOYEE relations that reference the tuple being

deleted from EMPLOYEE. We may enforce the constraint by: (i) rejecting the deletion, or

(ii) deleting all tuples in the WORKS\_ON, DEPENDENT, DEPARTMENT, and EMPLOYEE

relations whose values for ESSN, ESSN, MGRSSN, and SUPERSSN, respectively, is equal

to'987654321'

1. Delete the PROJECT tuple with Pname = ‘ProductX’.

Ans= Violates referential integrity because two tuples exist in the WORKS\_ON relations that

reference the tuple being deleted from PROJECT. We may enforce the constraint by: (i)

rejecting the deletion, or (ii) deleting the tuples in the WORKS\_ON relation whose value

for PNO=1 (the value for the primary key PNUMBER for the tuple being deleted from

PROJECT).

1. Modify the Mgr\_ssn and Mgr\_start\_date of the DEPARTMENT tuple with Dnumber = 5 to ‘123456789’ and ‘2007-10-01’, respectively.

Ans =No constraint violations.

1. Modify the Super\_ssn attribute of the EMPLOYEE tuple with Ssn = ‘999887777’ to ‘943775543’.

Ans = Violates referential integrity because the new value of SUPERSSN='943775543' and

there is no tuple in the EMPLOYEE relation with SSN='943775543'. We may enforce the

constraint by: (i) rejecting the deletion, or (ii) inserting a new EMPLOYEE tuple with

SSN='943775543'.

1. Modify the Hours attribute of the WORKS\_ON tuple with Essn = ‘999887777’ and Pno = 10 to ‘5.0’.

Ans=No constraint violations.

5.12

Consider the AIRLINE relational database schema shown in Figure 5.8, which describes a database for airline flight information. Each FLIGHT is identified by a Flight\_number, and consists of one or more FLIGHT\_LEGs with Leg\_numbers 1, 2, 3, and so on. Each FLIGHT\_LEG has scheduled arrival and departure times, airports, and one or more LEG\_INSTANCEs— one for each Date on which the flight travels. FAREs are kept for each FLIGHT. For each FLIGHT\_LEG instance, SEAT\_RESERVATIONs are kept, as are the AIRPLANE used on the leg and the actual arrival and departure times and airports. An AIRPLANE is identified by an Airplane\_id and is of a particular AIRPLANE\_TYPE. CAN\_LAND relates AIRPLANE\_TYPEs to the AIRPORTs at which they can land. An AIRPORT is identified by an Airport\_code. Consider an update for the AIRLINE database to enter a reservation on a particular flight or flight leg on a given date.

1. Give the operations for this update.

Ans= 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.What types of constraints would you expect to check?

Ans= I could 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.

1. Which of these constraints are key, entity integrity, and referential integrity constraints, and which are not?

Ans= 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).

1. Specify all the referential integrity constraints that hold on the schema shown in Figure 5.8.

Ans= We can 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\_INSTANC

5.16

Consider the following relations for a database that keeps track of student enrollment in courses and the books adopted for each course:

STUDENT(Ssn, Name, Major, Bdate)

COURSE(Course#, Cname, Dept) ENROLL(Ssn, Course#, Quarter, Grade)

BOOK\_ADOPTION(Course#, Quarter, Book\_isbn)

TEXT(Book\_isbn, Book\_title, Publisher, Author)

Specify the foreign keys for this schema, stating any assumptions you make.

Ans= The schema of this question has the following four foreign keys:

a. the attribute SSN of relation ENROLL that references relation STUDENT,

b. the attribute Course# in relation ENROLL that references relation COURSE,

c. the attribute Course# in relation BOOK\_ADOPTION that references relation COURSE, and

d. the attribute Book\_ISBN of relation BOOK\_ADOPTION that references relation TEXT.

5.17

Consider the following relations for a database that keeps track of automobile sales in a car dealership (OPTION refers to some optional equipment installed on an automobile):

CAR(Serial\_no, Model, Manufacturer, Price)

OPTION(Serial\_no, Option\_name, Price)

SALE(Salesperson\_id, Serial\_no, Date, Sale\_price)

SALESPERSON(Salesperson\_id, Name, Phone)

1. First, specify the foreign keys for this schema, stating any assumptions you make.
2. Next, populate the relations with a few sample tuples, and then give an example of an insertion in the SALE and SALESPERSON relations that violates the referential integrity constraints and of another insertion that does not.

Ans= a. The schema of this question has the following three foreign keys:

1. The attribute Serial\_no of relation OPTION that references relation CAR

2. The attribute Salesperson\_id of relation SALE that references relation SALESPERSON

3. The attribute Serial\_no of relation SALE that references relation CAR

Ans b. The foreign ids are Salesperson\_id and Serialno in SALE table.

VIOLATING INSERTION:-

INSERT INTO SALESPERSON(Salesperson\_id,'Name','Phone') VALUES(1,'Rajesh','9832511251');

INSERT INTO SALE(Salesperson\_id,Serialno,'Date','Sale\_price') VALUES(2,1,'2015-02-12','5000');

NON-VIOLATING INSERTION:-

INSERT INTO SALESPERSON(Salesperson\_id,'Name','Phone') VALUES(1,'Rajesh','9832511251');

INSERT INTO SALE(Salesperson\_id,Serialno,'Date','Sale\_price') VALUES(1,1,'2015-02-12','5000');

REASON:- The reason is that the foreign key value must be a subset of the primary key values. The primary key values set can be either same as the foreign key values set or a superset of it for referential integrity constraints to hold true.

5.18

Database design often involves decisions about the storage of attributes. For example, a Social Security number can be stored as one attribute or split into three attributes (one for each of the three hyphen-delineated groups of numbers in a Social Security number—XXX-XX-XXXX). However, Social Security numbers are usually represented as just one attribute. The decision is based on how the database will be used. This exercise asks you to think about specific situations where dividing the SSN is useful.

Ans= a. We need the area code (also know as city code in some countries) and perhaps the country code (for dialing international phone numbers).

b. I would recommend storing the numbers in a separate attribute as they have their own independent existence. For example, if an area code region were split into two regions, it would change the area code associated with certain numbers, and having area code in a separate attribute will make it is easier to update the area code attribute by itself.

c. I would recommend splitting first name, middle name, and last name into different attributes as it is likely that the names may be sorted and/or retrieved by the last name, etc.

d. In general, if the each attribute has an independent logical existence based on the application, it would make sense to store it in a separate column otherwise there is no clear advantage. For example, SSN need not be split into its component unless we are using the subsequences to make deductions about validity, geography, etc. In the two cases above, it made logical and business sense to split the attributes.