WEEK 2

CONSTRAINTS IN RELATIONAL DATABASE MODELS

STUDENT OBJECTIVES

- Upon completion of this video, you should be able to:
  - List key constraints in a table
  - Identify referential integrity constraints that are violated by insert, delete and modify actions on given tables
  - Identify semantic integrity constraints that are violated, given an existing table and an update operation on a table such as modify, insert or delete

TYPES OF CONSTRAINTS IN RELATIONAL DATABASES

- There are 3 main types of constraints:
  - Key constraints
  - Referential integrity constraints
  - Semantic integrity constraints

KEY CONSTRAINTS

- PRIMARY KEY - Allows you to state which attribute(s) will be the primary key (the attribute(s) that ensures that no 2 tuples are identical).
  - A table can only have ONE primary key but the primary key can be made up of several attributes
  - The primary key MUST be unique
- NOT NULL – Forces the user to never leave a key attribute null (empty) for a particular tuple.

REFERENTIAL INTEGRITY

Referential Integrity: a tuple in one relation (table) that refers to another relation (table) must refer to an existing tuple in the relation.

Formally:

Assume we have R1 and R2 with a referential integrity between the two of them. R1 has a set of attributes FK (foreign key) that references the attributes PK (primary key) R2, it must satisfy the following rules:

•FK attributes must have the same domain as PK

•a value of FK in a tuple t of the current state r1(R1) either occurs as a value of PK for some tuple t2 in the current state or r2(R2) is null.

UPDATE OPERATIONS ON RELATIONS MAINTAINING INTEGRITY RULES

Department

DeptID	DeptName	*MgrEmpID	MgrStartDate
G8H	Head Office	4	12/12/99
S7G	Safety Department	3	11/11/98
Y5J	Research Department	6	12/24/98

Employee

EmpID	LastName	FirstName	*DeptID	Sex
1	Simpson	Bart	S7G	M
2	Smithers	Waylan	G8H	M
4	Burns	Monty	G8H	M
6	Simpson	Lisa	Y5J	F
3	Beuvieau	Patty	S7G	M
12	Simpson	Homer	S7G	M

QUESTION: Determine the problems (if any exist) with the following operations to the above tables?

Insert Operation

IS THIS VALID?

Insert <13, 'Gumble', 'Barney', 'S7G', 'M'> into EMPLOYEE	✓Yes	No
Insert <3,'Simpson', 'Granpa', 'Y5J', 'M'> into EMPLOYEE	Yes	✓No
Insert <NULL, 'Flanders', 'Ned', 'Y5J', 'M'> into EMPLOYEE	Yes	✓No
Insert <18, 'Flanders', 'Todd', 'P68', 'M'> into EMPLOYEE	Yes	✓No

Department

DeptID	DeptName	*MgrEmpID	MgrStartDate
G8H	Head Office	4	12/12/99
S7G	Safety Department	3	11/11/98
Y5J	Research Department	6	12/24/98

Employee

EmpID	LastName	FirstName	*DeptID	Gender
1	Simpson	Bart	S7G	M
2	Smithers	Waylan	G8H	M
4	Burns	Monty	G8H	M
6	Simpson	Lisa	Y5J	F
3	Beuvieau	Patty	S7G	M
12	Simpson	Homer	S7G	M

Delete Operation

IS THIS VALID?

Delete employee where EmpID = 4 — Yes ✓No

Delete department where DeptID = 'S7G' — Yes ✓No

Department

DeptID	DeptName	*MgrEmpID	MgrStartDate
G8H	Head Office	4	12/12/99
S7G	Safety Department	3	11/11/98
Y5J	Research Department	6	12/24/98

Employee

EmpID	LastName	FirstName	*DeptID	Gender
1	Simpson	Bart	S7G	M
2	Smithers	Waylan	G8H	M
4	Burns	Monty	G8H	M
6	Simpson	Lisa	Y5J	F
3	Beuvieau	Patty	S7G	M
12	Simpson	Homer	S7G	M

QUESTION: DB2 allows 3 things to happen if you set up referential integrity between keys when you perform a delete, DB2 allows for:

- ***Cascade***
- ***Restrict***
- ***Set Null***

What do you think each of these operations do?

Modify Operation:

IS THIS VALID?

Modify the gender of Employee where lastname = ‘Burns' to ‘F’	✓Yes	No
Modify Employee where lastname = 'Smithers' from DeptID = 'G8H' to DeptID = 'Y5J'	✓Yes	No
Modify Employee where lastname = 'Smithers' from DeptID = 'G8H' to DeptID = 'J9J'	Yes	✓No
Modify Employee where lastname = 'Smithers' from EmpID = 2 to EmpID = 12	Yes	✓No

Department

DeptID	DeptName	*MgrEmpID	MgrStartDate
G8H	Head Office	4	12/12/99
S7G	Safety Department	3	11/11/98
Y5J	Research Department	6	12/24/98

Employee

EmpID	LastName	FirstName	*DeptID	Gender
1	Simpson	Bart	S7G	M
2	Smithers	Waylan	G8H	M
4	Burns	Monty	G8H	M
6	Simpson	Lisa	Y5J	F
3	Beuvieau	Patty	S7G	M
12	Simpson	Homer	S7G	M

SEMANTIC INTEGRITY CONSTRAINTS

- ***State Constraints***: state the constraints that a valid state of the database must satisfy

  Example: Hours worked cannot be greater than 50, Quantity Ordered must be greater than 10

- ***Transition Constraints***: define how the state of the database can change

  Example: Salaries can only increase

- Both of the above are enforce in relational databases through *triggers* and *assertions*

EXA

- Trigg
  does

Here is the trigger function befo_update:

```
01. CREATE OR REPLACE FUNCTION befo_update()
02.   RETURNS trigger AS
03. $$
04. BEGIN
05. NEW.TOTAL = NEW.SUB1 + NEW.SUB2 + NEW.SUB3 + NEW.SUB4 + NEW.SUB5;
06. NEW.PER_MARKS = NEW.TOTAL/5;
07. IF NEW.PER_MARKS >=90 THEN
08. NEW.GRADE = 'EXCELLENT';
09. ELSEIF NEW.PER_MARKS>=75 AND NEW.PER_MARKS<90 THEN
10. NEW.GRADE = 'VERY GOOD';
11. ELSEIF NEW.PER_MARKS>=60 AND NEW.PER_MARKS<75 THEN
12. NEW.GRADE = 'GOOD';
13. ELSEIF NEW.PER_MARKS>=40 AND NEW.PER_MARKS<60 THEN
14. NEW.GRADE = 'AVERAGE';
15. ELSE
16. NEW.GRADE = 'NOT PROMOTED';
17. END IF;
18.
19. RETURN NEW;
20. END;
21.
22. $$
23. LANGUAGE 'plpgsql';
```

Here is the trigger

```
01. CREATE TRIGGER updt_marks
02.   BEFORE UPDATE
03.   ON student_marks
04.   FOR EACH ROW
05.   EXECUTE PROCEDURE befo_update();
```

EXAMPLE OF A CONSTRAINT

- Constraint … s one does?

MySQL:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CHECK (Age>=18)
);
```

SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int CHECK (Age>=18)
);
```