Chapter 6: The Relational Data Model and Relational Database Constraints.

Outline:

6.1Relational Model Concepts
- Informal Definitions
- Formal Definitions
- Characteristics of Relations

6.2 Keys' Types
- Super key
- Key
- Primary Key
- Alternative Key

6.3 Relational Integrity Constraints
- Domain constraints
- Key constraints
- Entity integrity constraints
- Referential integrity constraints
- Semantic Integrity Constraints

6.4 Update Operations and Dealing with Constraint Violations
- Insert Operation
- Delete Operation
- Update Operation

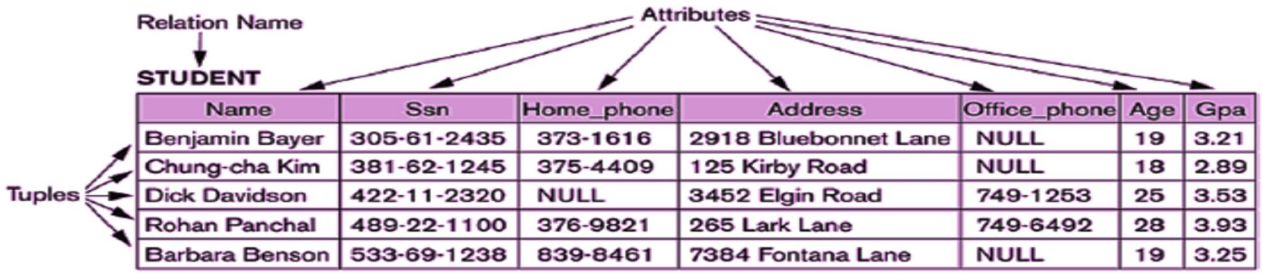
6.1 Relational Model Concepts

- The relational Model of Data is based on the concept of a Relation.
- A Relation is a mathematical concept based on the ideas of sets.
- The model was first introduced by Tod Codd of IBM Research in 1970 in the following paper: "A Relational Model for Large Shared Data Banks," Communications of the ACM, June 1970.

Informal Definitions

- Informally, a relation looks like a table of values.
- A relation typically contains a set of rows.
- The data elements in each row represent certain facts that correspond to a real-world entity or relationship
  - In the formal model, rows are called tuples
- Each column has a column header that gives an indication of the meaning of the data items in that column
  - In the formal model, the column header is called an attribute

Example of a Relation



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	749-1253	25	3.53
Rohan Panchal	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
Barbara Benson	533-69-1238	839-8461	7384 Fontana Lane	NULL	19	3.25

Formal Definitions

1. ***Schema***
   - The Schema (or description) of a Relation:
     - Denoted by R(A1, A2, ...An)
     - R is the name of the relation
     - The attributes of the relation are A1, A2, ..., An
   - ***Example:***
     CUSTOMER (Cust-id, Cust-name, Address, Phone#)
     - CUSTOMER is the relation name
     - Defined over the four attributes: Cust-id, Cust-name, Address, Phone#
   - Each attribute has a domain or a set of valid values.
   - For example, the domain of Cust-id is 6 digit numbers.
   - The data domain may have a data-type and/or format.

2. *Tuple*

- A tuple is an ordered set of values (enclosed in angled brackets '< … >')
- Each value is derived from an appropriate domain.
- A *tuple* can be considered as a set of (<attribute>, <value>) pairs. Thus the following two tuples are *identical*:

```
t1 = <(Name, B. Bayer),(SSN, 305-61-2435),(HPhone,
3731616),(Address, 291 Blue Lane),(WPhone, null),(Age,
23),(GPA, 3.25)>

t2 = <(HPhone, 3731616),(WPhone, null),(Name, B. Bayer),(Age,
23),(Address, 291 Blue Lane),(SSN, 305-61-2435),(GPA, 3.25)>
```

3. *State*

- The relation state is a subset of the Cartesian product of the domains of its attributes
- Each domain contains the set of all possible values the attribute can take.
- Example: attribute Cust-name is defined over the domain of character strings of maximum length 25

  dom(Cust-name) is varchar(25)

4. *Summary*

Informal Terms	Formal Terms
Table	Relation
Column Header	Attribute
All possible Column Values	Domain
Row	Tuple
Table Definition	Schema of a Relation
Populated Table	State of the Relation

Table 6.1: Formal / Informal Summary

Characteristics of Relations:

1. Ordering of tuples in a relation r(R): The tuples are not considered to be ordered, even though they appear to be in the tabular form.
2. Ordering of attributes in a relation schema R (and of values within each tuple): We will consider the attributes in R(A1, A2, ...,An) and the values in t=<v1, v2, ..., vn> to be ordered.
3. Values in a tuple:
   - All values are considered atomic (indivisible).
   - Each value in a tuple must be from the domain of the attribute for that column
     - If tuple t = <v1, v2, …, vn> is a tuple (row) in the relation state r of R(A1, A2, …, An)

- Then each vi must be a value from dom(Ai)
- A special null value is used to represent values that are unknown or unapplicable to certain tuples.

6.2 Keys' Types

- In order to understand the Relational Integrity Constraints we need to understand the Types of keys before
  1. ***Superkey of R (SK)***
     - SK is a set of attributes of R with the following condition:
       - No two tuples in any valid relation state r(R) will have the same value for SK
       - That is, for any distinct tuples t1 and t2 in r(R), t1[SK] ≠ t2[SK]
  2. ***Key of R***
     - A "minimal" superkey
     - That is, a key is a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey (does not possess the superkey uniqueness property)
     - Each of the keys is called **candidate key**
  3. ***Primary key (PK)***
     - Choice **one** of the candidate key
     - It is usually better to choose a primary key with a single attribute or a small number of attributes.
  4. ***Alterative key***
     - all candidate keys except primary key

Example (1):
Consider the EMPLOYEE relation schema:
Employee (SSN,Empid,Fname,Mname,Lname,Add,Age)

1. **Superkey of R:**
   {SSN},{Empid},{SSN,Empid},{SSN,Fname},{SSN,Mname},{SSN,Lname},{ SSN,Add} , {SSN,Age},{ SSN,Empid,Fname},{ SSN,Empid,Lname},{ SSN,Empid,Mname},{ SSN,Empid,Add},{ SSN,Empid,Age},{Fname, Mname, Lname},…etc.
2. **Key of R:**
   {SSN}, {Empid},{Fname,Mname,Lname}
3. **Primary key:**
   {SSN}
4. **Alterative key:**
   {Empid},{Fname,Mname,Lname}

Example (2):
Consider the following relation:

X	Y	Z
1	A	20
3	B	21
4	A	15
1	C	30

1. **Superkey of R:**
{Z},{X,Y},{X,Z},{Y,Z},[X,Y,Z}
2. **Key of R:**
{Z}, {X, Y}
3. **Primary key:**
{Z}
4. **Alterative key:**
{X, Y}

Find superkeys, candidate keys, primary keys, alternative keys in the following CAR relation schema?

CAR (State, Reg#, SerialNo, Model, Year)

Find superkeys, candidate keys, primary keys, alternative keys in the following relation?

A	B	C	D	E
A1	Hh	101	D10	10
A2	Mm	200	D30	20
A3	Ff	50	D90	30
A4	Ll	60	D53	30
A1	Ff	70	D6	10

6.3 Relational Integrity Constraints

- **Constraints** are conditions that must hold on all valid relation states.

- There are five types of constraints in the relational model:
  1. ***Domain constraints***
  2. ***Key constraints***
  3. ***Entity integrity constraints***
  4. ***Referential integrity constraints***
  5. ***Semantic Integrity Constraints***

Domain Constraints

- Every value in a tuple must be from the domain of its attribute (or it could be null, if allowed for that attribute). A data type or format is also specified for each domain.
- Special case of this constraint ***Constraints on Null*** (not allowed null values for attribute)

Key Constraints

- For any two distinct tuples t_1 and t_2 in a relation state *r* of *R* we have $t1[SK] \neq t_2[SK]$ (the subset of attributes *SK* is called a *superkey* of the relation schema *R*).
- A superkey *SK* specifies a uniqueness constraint.

Entity Integrity

- The primary key attributes PK of each relation schema R in S cannot have null values in any tuple of r(R). $t[PK] \neq null$ for any tuple t in r(R)
  - This is because primary key values are used to identify the individual tuples.
  - If PK has several attributes(composite), null is not allowed in any of these attributes

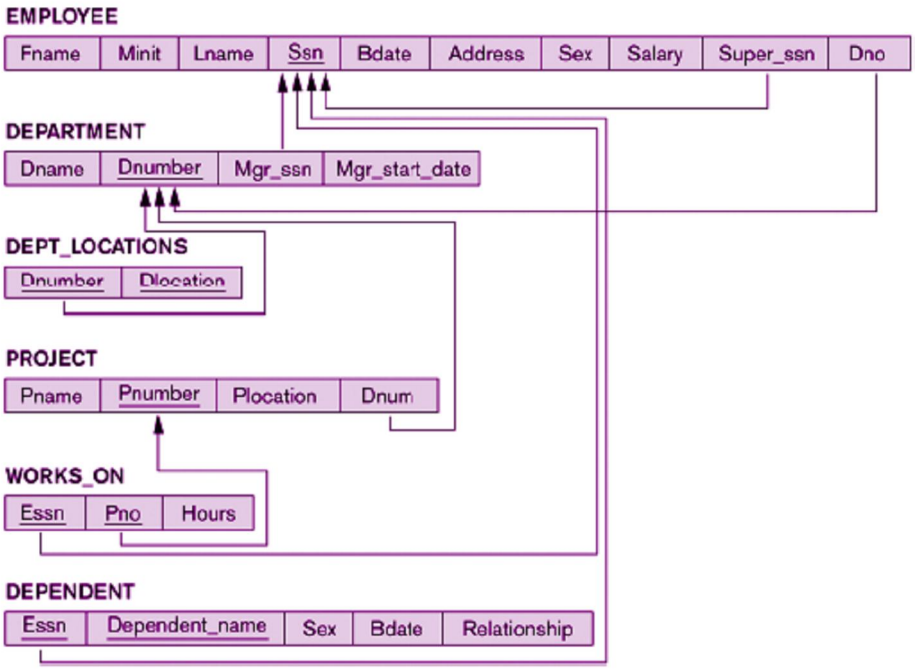
Referential Integrity

- A constraint involving *two relations* (the previous constraints involve a *single* relation). Used to specify a *relationship* among tuples in two relations: the **referencing relation** and the **referenced relation**.
- Referential Integrity is used to maintain the consistency among tuples of the two relations.
- The referential integrity constraint states that a tuple in one relation that refers to another relation must refer to an existing tuple in that relation.
- Tuples in the *referencing relation* R1 have attributes FK (called **foreign key** attributes) that reference the primary key attributes PK of the *referenced relation* R2. A tuple t_1 in R_1 is said to **reference** a tuple t_2 in R_2 if

$$t_1[FK] = t_2[PK]$$

- A referential integrity constraint can be displayed in a relational database schema as a directed arc from R_1 FK to R_2 PK.

Referential integrity constraints displayed on the COMPANY relational database schema.



- The value in the foreign key column (or columns) FK of the referencing relation R1 can be either:
    1. A value of an existing primary key value of acorresponding primary key PK in the referenced relation R2, or
    2. A null (if it is not part of its own primary key)

Semantic Integrity Constraints

- Based on application semantics and cannot be expressed by the model.
- Example1: the max. no. of hours per employee for all projects he or she works on is 56 hrs per week.
- Example2: the salary of an employee should not exceed the salary of the employee's supervisor.
- SQL-99 allows triggers and ASSERTIONS to express for some of these

6.4 Update Operations and Dealing with Constraint Violations

- Operations of the relational model:

1. Updates:
    - *Insert* - to insert a new tuple or tuples in a relation,
    - *Delete* - to delete tuples from a relation, and
    - *Update* (or *Modify*) - to change the values of some attributes in existing tuples.

2. Retrievals - these operations do not change the current state of the databases.

- Integrity constraints should not be violated by the update operations: INSERT, DELETE, MODIFY
- In **case of integrity violation**, several actions can be taken, such as:
    - Cancel the operation that causes the violation (RESTRICT or REJECT option)
    - Trigger additional updates so the violation is corrected (CASCADE option, SET NULL option)

Insert Operation

- ***INSERT may violate any of the constraints:***

    - Domain constraint: if one of the attribute values provided for the new tuple is not of the specified attribute domain.
    - Key constraint: if the value of a key attribute in the new tuple already exists in another tuple in the relation
    - Entity integrity: if the primary key value is null in the new tuple
    - Referential integrity: if a foreign key value in the new tuple references a primary key value that does not exist in the referenced relation

Delete Operation

- DELETE may violate only referential integrity: If the primary key value of the tuple being deleted is referenced from other tuples in the database
- Can be remedied by several actions: RESTRICT, CASCADE ,SET NULL
    - *RESTRICT option*: reject the deletion
    - *CASCADE option*: delete the tuples that reference the tuple that is being deleted
    - *SET NULL option*: set the foreign keys of the referencing tuples to NULL
- One of the above options must be specified during database design for each foreign key constraint

Update Operation

- It may violate any of the constraints.

Which **relational constraints** would be **violated** if the following operation were applied to the COMPANY database snapshot illustrated bellow?

FName	LName	SSN	BDate	Salary	SuperSSN	DNo
Smith	Jone	123	1/2/1979	2100JD	NULL	10
Susan	Adams	432	3/6/1982	1547JD	123	10
James	Ford	587	20/9/1980	3650JD	983	20
Sali	Miller	983	1/12/1970	890JD	123	30

Department

DNo	DName
10	Headquarter
20	Accounting
30	IT

Insert

2. Insert < 'Jasmine', 'Rani', 321, 3/9/1982, 3000, NULL, 30> into EMPLOYEE.

3. Insert < 'Jasmine', 'Rani', 923, 3/19/1980, '3000JD', NULL, 30> into EMPLOYEE.

4. Insert <123456, 'Rani', 321, 3/9/1985, '3000$', 432, 10> into EMPLOYEE.

5. Insert <400,'Planning'> into DEPARTMENT.

6. Insert <123456, 'Rani', 123, 3/9/1985, '3500JD', 432, 10> into EMPLOYEE.

7. Insert <30,'Planning'> into DEPARTMENT.

8. Insert < 'Jasmine', 'Rani',NULL, 30/9/1977, '4000JD', 587, 10> into EMPLOYEE.

9. Insert <NULL,'Planning'> into DEPARTMENT.

10. Insert < 'Jasmine', 'Rani',222, 3/2/1980, 3000, 587, 50> into EMPLOYEE .

11. Insert < 'Alice', 'Rani',222, 30/2/1976, 3000, 687, 10> into EMPLOYEE.

12. Insert < 'Jasmine', 'Rani',NULL, 3/22/1976, '1000JD', 587, 50> into EMPLOYEE.

13. Insert < 'Sali', 'Rani', 'NULL', 3/2/1969, '6000JD', 587, 40> into EMPLOYEE.

14. Insert < 'Jasmine', 'Rani', 123, 3/2/76, '3200JD', 587, 60> into EMPLOYEE.

Delete

1. Delete the DEPARTMENT tuple with DNo =20

2. Delete the EMPLOYEE tuple with FName ='Smith'

3. Delete the EMPLOYEE tuple with SSN=432

Modify

1. Update the SALARY of the EMPLOYEE tuple with SSN=432 to 2800.

2. Update the DNO of the EMPLOYEE tuple with SSN=432 to 70.

3. Update the DNO of the DEPARTMENT tuple with DName ='IT' to 40.

4. Update the DNO of the DEPARTMENT tuple with DName ='IT' to 10.

5. Update the SSN of the EMPLOYEE tuples with FName ='Smith' to NULL.

6. Update the SSN of the EMPLOYEE tuples to 566.