International Institute of Information Technology Bangalore

The Relational Data Model and Constraints

CS 301

“Goal of the chapter”

“Learn about the basic principles of the relational data model.”

1. Define the modeling concepts and notations of the relational model.

2. Discuss relational constraints and relational database schemas.

3. Define the update operations of the relational model and handling the violations of integrity constraints.

Relational model concepts

- The relational model represents DB as a collection of "*relations*" (resembles a table of values, a flat file of records – because each record has a simple linear or flat structure).

- In a table, each row is a collection of related values. Table name and column names are used to interpret the meanings of the values in each row. Example, STUDENT DB. All values in a column are of the same data type.

- Some relations represent facts about "entities", whereas others represent facts about "relationships".

- Row → tuple
- Column header → attribute
- Table → relation
- Data type of each column → represented by a domain of possible values.

Definitions

- Generally, 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 name** (or just **attribute**).

Tuple

- A **tuple** is an ordered set of values (enclosed in angled brackets '< … >').

- Each value is derived from an appropriate *domain*.

- A row in the CUSTOMER relation consist of four values, for example:

A tuple (row) in the CUSTOMER relation.

- ✓ <632895, "John Smith", "101 Main St. Atlanta, GA 30332", "(404) 894-2000">
- ✓ This has 4 values.

- A relation is a **set** of such tuples (rows).

Domain

- A domain *D* is a set of atomic values (each value is indivisible) specified by a name, data type or format (character string, numeric, integer, etc.).
  - Examples of domains are:
    - Names: set of characters that represent name of a person.
    - Grade_point_average: Possible floating-point number between 0 and 4.
    - Employee_age: Possible age of employees in a company between 18 and 60.
- Additional information can also be given such as Person_weights with units such as pounds or kilograms.

Domain

- A **domain** has a logical definition:
  - Example: "Indian_mobile_numbers" are the set of 10 digit phone numbers valid in India.

- A domain format
  - The Indian_mobile_numbers may have a format: *nn-nnnn-nnnn* where each "*n*" is a numeric (decimal) digit.
  - Dates have various formats such as year, month, date formatted as yyyy-mm-dd, or as dd mm, yyyy, etc.

- The attribute name designates the role played by a domain in a relation:
  - Used to interpret the meaning of the data elements corresponding to that attribute.
  - Example: The domain "Date" may be used to define two attributes named "Invoice-date" and "Payment-date" with different meanings.

Relational model concepts

- A Relation is a mathematical concept based on the ideas of sets.
- The model was first proposed by Dr. E. F. Codd of IBM Research in 1970 in the following paper:
  - "A Relational Model for Large Shared Data Banks," *Communications of the ACM*, June 1970.
- The above paper caused a major revolution in the field of database management and earned Dr. Codd the coveted ACM Turing Award.

Relation schema

- The **Schema** (or description) of a Relation *R* is denoted by $R(A_1, A_2, ..., A_n)$.
  - *R* is the **name** of the relation.
  - The **attributes** of the relation are $A_1, A_2, ..., A_n$.

A **relation schema** R, denoted by $R(A_1, A_2, \ldots, A_n)$, is made up of a relation name R and a list of attributes, $A_1, A_2, \ldots, A_n$. Each **attribute** A_i is the name of a role played by some domain D in the relation schema R. D is called the **domain** of A_i and is denoted by **dom**(A_i). A relation schema is used to *describe* a relation; R is called the **name** of this relation. The **degree** (or **arity**) of a relation is the number of attributes n of its relation schema.

A relation of degree seven, which stores information about university students, would contain seven attributes describing each student as follows:

STUDENT(Name, Ssn, Home_phone, Address, Office_phone, Age, Gpa)

Using the data type of each attribute, the definition is sometimes written as:

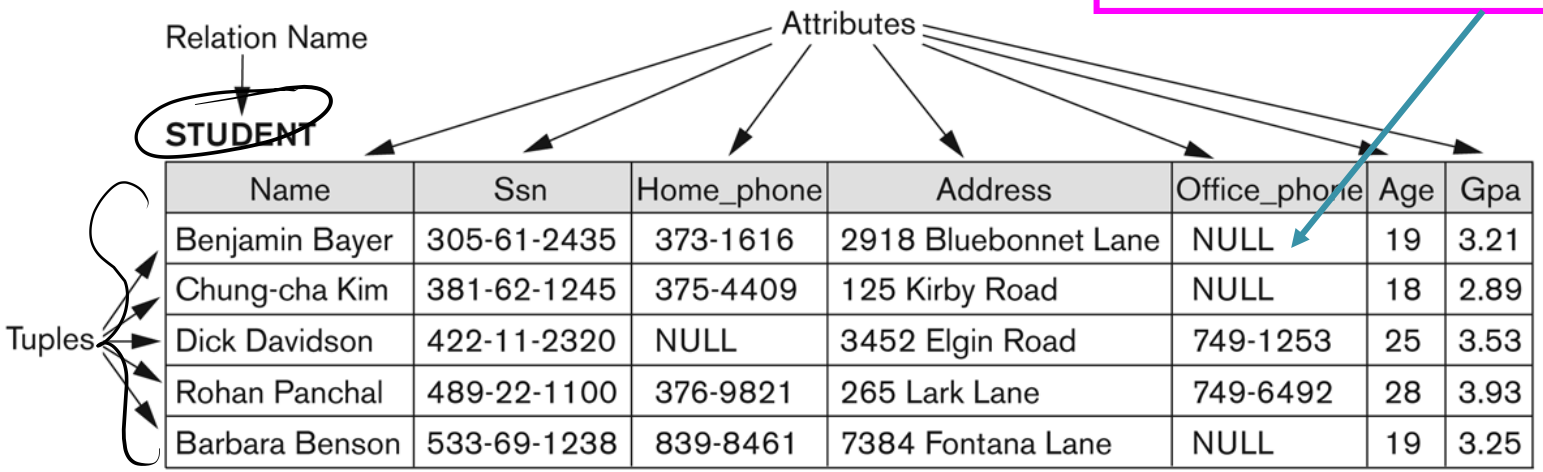
STUDENT(Name: string, Ssn: string, Home_phone: string, Address: string, Office_phone: string, Age: integer, Gpa: real)

Example of a relation

A **relation** (or **relation state**) r of the relation schema $R(A_1, A_2, \ldots, A_n)$, also denoted by $r(R)$, is a set of n-tuples $r = \{t_1, t_2, \ldots, t_m\}$. Each n-**tuple** t is an ordered list of n values $t = <v_1, v_2, \ldots, v_n>$, where each value v_i, $1 \leq i \leq n$, is an element of dom (A_i) or is a special NULL value.

The ith value in tuple t, which corresponds to the attribute A_i, is referred to as $t[A_i]$ or $t.A_i$ (or $t[i]$ if we use the positional notation). The terms **relation intension** for the schema R and **relation extension** for a relation state $r(R)$ are also commonly used.

NULL – values are unknown or do not exist.



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

The attributes and tuples of a relation STUDENT.

Relation

- Relation can be stated using set theory.

A relation (or relation state) $r(R)$ is a **mathematical relation** of degree n on the domains $\text{dom}(A_1), \text{dom}(A_2), \ldots, \text{dom}(A_n)$, which is a **subset** of the **Cartesian product** (denoted by $\times$) of the domains that define R:

$$r(R) \subseteq (\text{dom}(A_1) \times \text{dom}(A_2) \times \ldots \times (\text{dom}(A_n))$$

The Cartesian product specifies all possible combinations of values from the underlying domains. Hence, if we denote the total number of values, or **cardinality,** in a domain D by $|D|$ (assuming that all domains are finite), the total number of tuples in the Cartesian product is

$$|\text{dom}(A_1)| \times |\text{dom}(A_2)| \times \ldots \times |\text{dom}(A_n)|$$

- This product of cardinalities of all domains represents the total number of possible instances or tuples that can ever exist in any relation state *r(R)*.
- One of these possible combinations at a given time - is the current relation state.
- The attribute names indicate different roles for the domain.

True or False

Relation state remains static whereas schema *R* changes occasionally.

False

True: Relation state change frequently whereas schema *R* is relatively static.

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.

- For example: attribute "Cust-name" is defined over the domain of character strings of maximum length 25.
  - dom(Cust-name) is varchar(25)

- The role these strings play in the CUSTOMER relation is that of the "*name of a customer*".

Example

- Let $R(A_1, A_2)$ be a relation schema:
  - Let dom(A_1) = {0,1}
  - Let dom(A_2) = {a,b,c}
- Then: dom(A_1) X dom(A_2) is all possible combinations:

  {<0, a> , <0, b> , <0, c>, <1, a>, <1, b>, <1, c> }

- The relation state $r(R) \subset$ dom(A_1) X dom(A_2)

- For example: $r(R)$ could be {<0, a> , <0, b> , <1, c> }
  - this is one possible state (or "population" or "extension") r of the relation R, defined over A_1 and A_2.
  - It has three 2-tuples: <0, a> , <0, b> , <1, c>

Summary

- Formally,
  - Given $R(A_1, A_2, ..., A_n)$
  - $r(R) \subset \text{dom}(A_1) \times \text{dom}(A_2) \times ....\times \text{dom}(A_n)$

- $R(A_1, A_2, \ldots, A_n)$ is the **schema** of the relation.

- R is the **name** of the relation

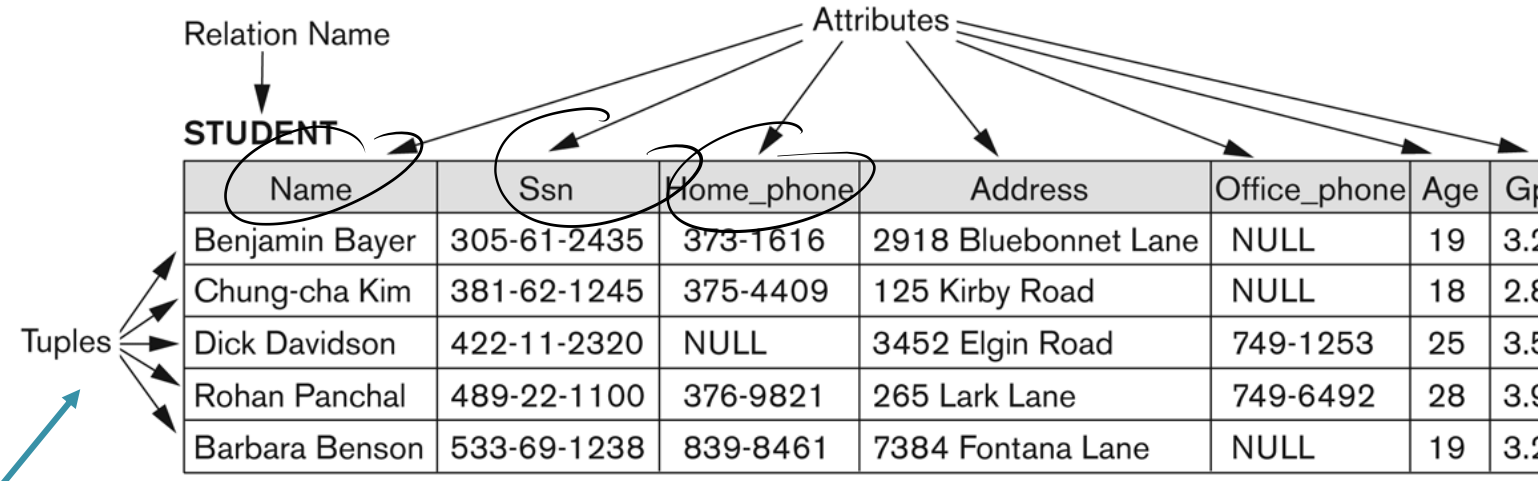
- $A_1, A_2, \ldots, A_n$ are the **attributes** of the relation.

- $r(R)$: a specific **state** (or "value" or "population") of relation R – this is a *set of tuples* (rows)
  - $r(R) = \{t_1, t_2, \ldots, t_n\}$ where each t_i is an n-tuple
  - $t_i = \langle v_1, v_2, \ldots, v_n \rangle$ where each v_i is an *element-of* $\text{dom}(A_j)$

Characteristics of relations

- 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, OR relation is not sensitive to the ordering of tuples.
  - *The definition of a relation does not specify any order*.

Example – A relation STUDENT



STUDENT

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

The attributes and tuples of a relation STUDENT.

Both are identical

STUDENT

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

The relation STUDENT with a different order of tuples.

Characteristics of relations

- Ordering of attributes in a relation schema *R* (and of values within each tuple) is important:
  - We will consider the attributes in $R(A_1, A_2, ..., A_n)$ and the values in $t=<v_1, v_2, ..., v_n>$ to be ordered.
  - However, a more "general alternative definition" of relation does not require this ordering because a tuple can be considered as a set of (<attribute>, <value>) pairs. The below two tuples are identical and are called self-describing data.

t = < (Name, Dick Davidson),(Ssn, 422-11-2320),(Home_phone, NULL),(Address, 3452 Elgin Road),
(Office_phone, (817)749-1253),(Age, 25),(Gpa, 3.53)>

t = < (Address, 3452 Elgin Road),(Name, Dick Davidson),(Ssn, 422-11-2320),(Age, 25),
(Office_phone, (817)749-1253),(Gpa, 3.53),(Home_phone, NULL)>

Two identical tuples when the order of attributes and values is not part of relation definition.

NULL values in tuples

Quiz

- There are three meanings of NULL –

1. value unknown
2. value exist but is not available
3. attribute does not apply (or value undefined)

NULL values in tuples

- The exact meaning of NULL value (either value unknown, or value exist but is not available, or attribute does not apply) "governs" how it fares during arithmetic aggregations or comparison.
  - Example, a comparison of two NULL values – if both customer *A* and *B* have NULL address, it does not mean that they have the same address.
- It is best "*to avoid NULL values*" as much as possible.

Definition 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

Relational model constraints and relational DB schemas

- There are restrictions or constraints on the DB state which are derived from the rules of the world.

- Constraints on the DB can be divided into three main categories:

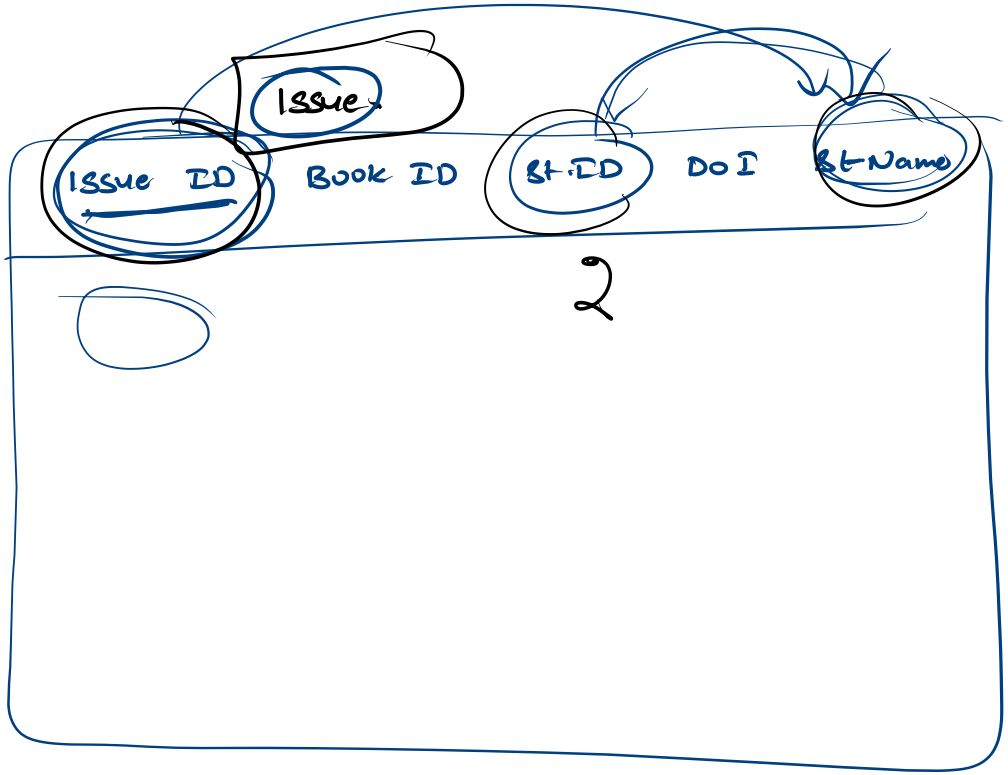
1. Inherent model-based constraints (implicit constraints) – that are inherent in the data model.
2. Schema-based constraints (explicit constraints) – they can be directly expressed in the schemas of the data model using DDL. Example, domain constraints, key constraint, constraints on NULL, entity integrity constraints, and referential integrity constraints.
3. Application based (semantic or business rules) – they *cannot* be directly expressed in the schema and are expressed and enforced by the application program. These are specified as **assertions** in SQL.

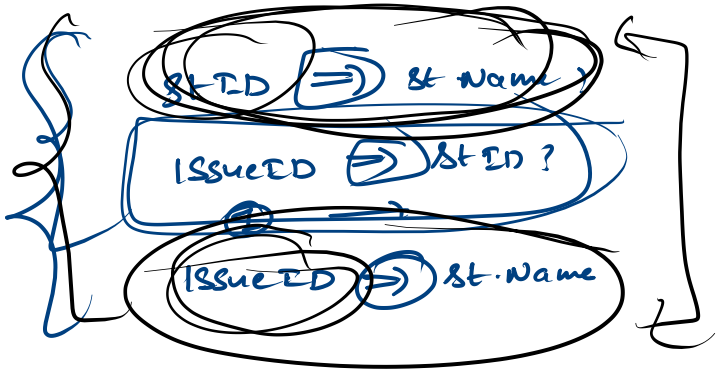
- Another important category of constraints is data dependencies that includes functional dependencies and multivalued dependencies (we will discuss more on these two dependencies during normalization of relations at a later part of the course).

- These are used for testing the "goodness" of the design of a relational DB and are utilized in normalization.

- The schema based constraints are domain constraints, key constraints, constraints on NULL, entity integrity constraints, and referential integrity constraints.

Let's see them (one by one) …





A ⇏ B. !

BookID ⇒ St.ID

Domain constraints

- They specify that within each tuple, the value of each attribute must be atomic from the domain dom*(A)*.

- Domains have a data type (such as numeric, integer, real number, Characters, Booleans, fixed and variable-length strings, date, time, timestamp and other special data types) and a range of values.

Next, we will discuss key constraints and constraints on NULL values.

Domain constraints

- All tuples in a relation must be distinct – i.e. no two tuples can have the same combination of values for all their attributes.

- Usually, there are subsets of attributes with the property that no two tuples have the same combination of values – such set of attributes is called **superkey – it specifies *uniqueness constraint***.

- A relation is a set of tuples and all elements of a set are distinct, therefore, all tuples in a relation must also be distinct.
- This means that no two tuples can have the same combination of values for all their attributes.

- There are **subset of attributes** of a relation schema *R* with the property that no two tuples should have the same combination of values for these attributes.
- Any such set of attributes is called a "***superkey***" of the relation schema R.
- A **superkey** specifies a ***uniqueness constraint*** that no two distinct tuples in any state *r* of *R* can have the same value.

- Every relation has at least one default superkey – the set of all its attributes.

What is a key?

candidate key

- "Key" of a Relation is a superkey of *R* with the additional property that removing any attribute leave a set of attribute that does not lead to superkey.

  - Each row has a value of a data item (or set of items) that uniquely identifies that row in the table – **this is *key***.

  - In the STUDENT table, SSN/Aadhar is the key.

  - Sometimes row-ids or sequential numbers are assigned as keys to identify the rows in a table – this is *artificial key* or *surrogate key*.

Key constraints

A key satisfies two properties:

- **Superkey** of *R*:
  - is a set of attributes SK of *R* with the following condition:
    - No two tuples in any valid relation state *r(R)* will have the same (identical) values for SK (*uniqueness property*).
    - That is, for any distinct tuples t_1 and t_2 in *r(R)*, $t_1[SK] \neq t_2[SK]$.
    - This condition must hold in *any valid state r(R)*.

- **Key** of *R*:
  - A "minimal" superkey.
  - That is, a key is a superkey *K* such that removal of any attribute *A* from *K* results in a set of attributes *K´* that is not a superkey of *R* any more (does not possess the superkey uniqueness property).
  - This is called a *minimality property* – means a superkey from which we cannot remove any attribute and still have the "*uniqueness constraint*" hold. *It is required for a key but is optional for a superkey.*

True or False?

- A key is a superkey but not vice versa.

True

- A superkey may be a key (if it is minimal) or may not be a key (if it is not minimal).

True

Key constraints

- If a relation schema has more than one key then each of the keys is a "**candidate key**" (License_number and Engine_serial_number in relation CAR below), one is chosen arbitrarily to be the "**primary key**". The other candidate keys are "unique keys".

CAR

License_number	Engine_serial_number	Make	Model	Year
Texas ABC-739	A69352	Ford	Mustang	02
Florida TVP-347	B43696	Oldsmobile	Cutlass	05
New York MPO-22	X83554	Oldsmobile	Delta	01
California 432-TFY	C43742	Mercedes	190-D	99
California RSK-629	Y82935	Toyota	Camry	04
Texas RSK-629	U028365	Jaguar	XJS	04

- The primary key attributes are underlined. The primary key value is used to *uniquely identify* each tuple in a relation. It provides the tuple identity.

- Primary key is also used to *reference* the tuple from another tuple.
  - General rule: Choose a primary key which is the smallest of the candidate keys (in terms of size / with a single or a small number of attributes).
  - Not always applicable – choice is sometimes subjective.

Key constraints

- Example: Consider the CAR relation schema:
  - CAR(License_number, Engine_serial_number, Make, Model, Year)
  - CAR has two keys:
    - Key1 = {License_number}
    - Key2 = {Engine_serial_number}
  - Both are also superkeys of CAR
  - {Engine_serial_number, Make} is a superkey but *not* a key.

- In general:
  - Any *key* is a *superkey* (but not vice versa).
  - Any set of attributes that *includes a key* is a *superkey*.
  - A *minimal* superkey is also a key.

CAR table with two candidate keys – License_number chosen as Primary Key

CAR

License_number	Engine_serial_number	Make	Model	Year
Texas ABC-739	A69352	Ford	Mustang	02
Florida TVP-347	B43696	Oldsmobile	Cutlass	05
New York MPO-22	X83554	Oldsmobile	Delta	01
California 432-TFY	C43742	Mercedes	190-D	99
California RSK-629	Y82935	Toyota	Camry	04
Texas RSK-629	U028365	Jaguar	XJS	04

The CAR relation, with two candidate keys: License_number and Engine_serial_number.

- Another constraint on attributes specifies whether NULL values are permitted or not.

- Example, if every STUDENT tuple must have a valid, non-NULL value for the Name attribute, then Name of STUDENT is constrained to be "NOT NULL".

Relational integrity constraints (IC)

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

- There are three *main types* of constraints in the relational model:
  1. **Key** constraints (Refer slide 28 to 31)
  2. **Entity integrity** constraints
  3. **Referential integrity** constraints

- Another implicit constraint is the **domain** constraint.
  - Every value in a tuple must be from the *domain of its attribute* (or it could be NULL, if allowed for that attribute).

Relational DB and Schemas

- Relational DB schema is
  - $S = \{R_1, R_2, ..., R_m\}$, a set of relation schemas that belong to the same database.
  - $R_1, R_2, \ldots, R_m$ are the names of the individual relation schemas within the relational database schema S.

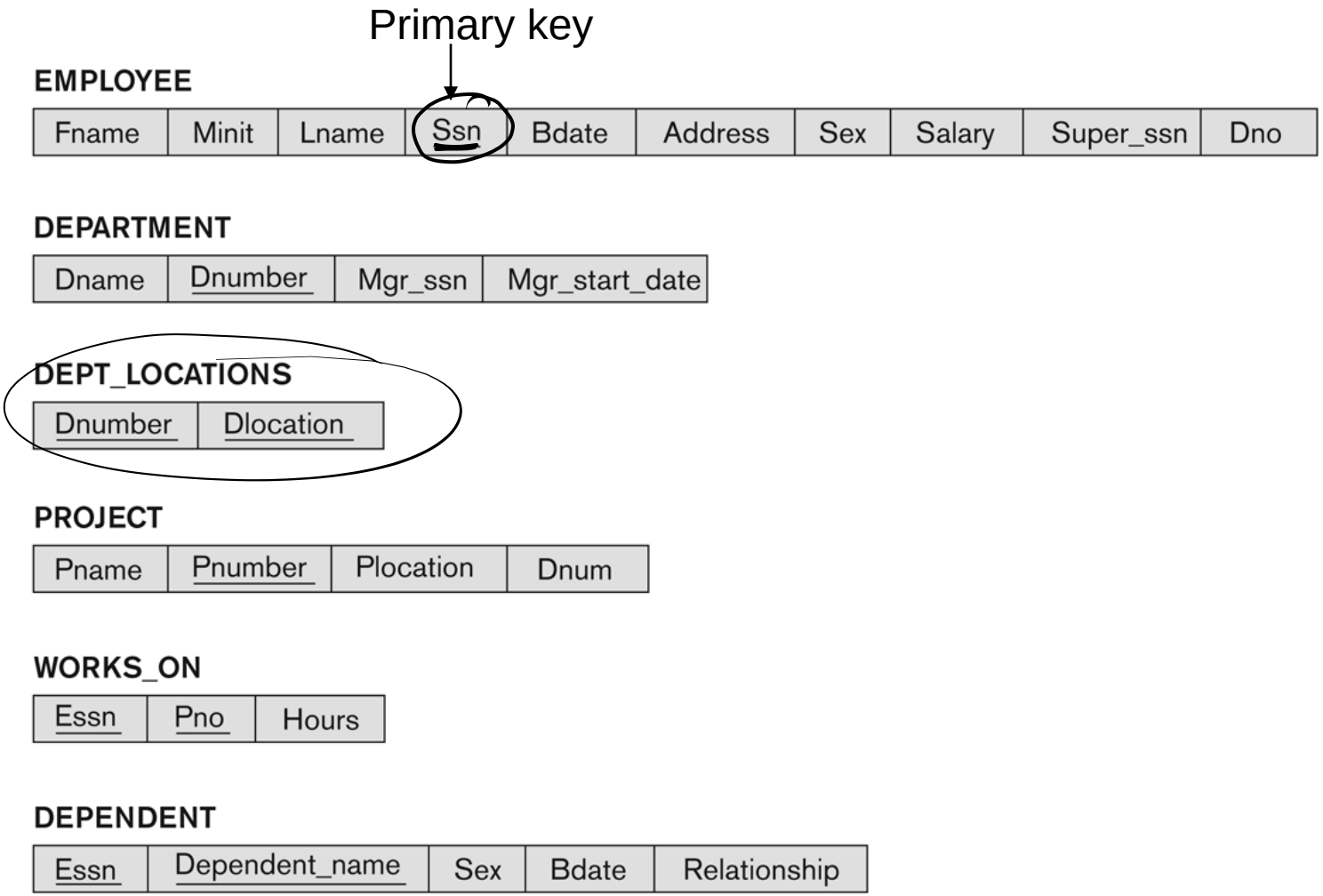
In other words,

A **relational database schema** S is a set of relation schemas $S = \{R_1, R_2, \ldots, R_m\}$ and a set of **integrity constraints** IC. A **relational database state** DB of S is a set of relation states DB $= \{r_1, r_2, \ldots, r_m\}$ such that each r_i is a state of R_i and such that the r_i relation states satisfy the integrity constraints specified in IC.

Let's see an (old) example …

COMPANY database schema

COMPANY database schema with 6 relation schemas.



Schema diagram for the COMPANY relational DB schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

One possible DB state for the COMPANY relational DB schema.

- A DB state that satisfies all the constraints in the defined set of integrity constraints (IC) is called a *valid state*.
- A DB state that does not obey all the integrity constraints is called *not valid*.

Therefore,

- Integrity constraints are specified on a DB schema and are expected to hold on every valid DB state of that schema.
- Next, we will see entity integrity and referential integrity.

Entity Integrity

- **Entity Integrity:**
  - The *primary key attributes PK* of each relation schema *R* in S *cannot have NULL values* in any tuple of *r(R)* – *i.e. no primary key value can be NULL*.

    - This is because primary key values are used to *identify* the individual tuples. If two or more tuples have NULL for their primary keys, we cannot identify them.
    - *t*[PK] ≠ NULL for any tuple *t* in *r(R)*.
    - If PK has several attributes, NULL is not allowed in any of these attributes.

  - Note: Other attributes of *R* may be constrained to disallow NULL values, even though they are not members of the primary key. Key constraints and entity constraints are specified on individual relations.

Referential Integrity

- It is a constraint specified "between two relations", and is used to maintain the consistency among tuples in the two relations.
- Example, attribute "Dno" of EMPLOYEE gives the department number for which each employee works; its value in every EMPLOYEE tuple must match the "Dnumber" value of some tuple in the DEPARTMENT relation.
  - One is a referencing relation and the other is a referenced relation.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

Referential Integrity

- A set of attributes FK in relation schema R_1 is a foreign key of R_1 that references relation R_2 if it satisfies the following rules:

1. The attributes in FK have the same domain(s) as the primary key attributes PK of R_2; the attributes FK are said to **reference** or **refer to** the relation R_2.
2. A value of FK in a tuple t_1 of the current state $r_1(R_1)$ either occurs as a value of PK for some tuple t_2 in the current state $r_2(R_2)$ *or is NULL*. In the former case, we have $t_1[\text{FK}] = t_2[\text{PK}]$, and we say that the tuple t_1 **references** or **refers to** the tuple t_2.

- R_1 is called the referencing relation and R_2 is the referenced relation. If these two conditions hold, a **referential integrity** constraint from R_1 to R_2 is said to hold.

- Referential integrity constraints typically arise "from the relationship" among the entities represented by the relation schemas.

Referential Integrity

- Tuples in the **referencing relation** R_1 have attributes **FK** (called **foreign key** attributes) that reference the **PK** (called **primary key** attributes) of the **referenced relation** R_2.
  - A tuple t_1 in R_1 is said to **reference** a tuple t_2 in R_2 if $t_1[FK] = t_2[PK]$.

- Example, Dno from EMPLOYEE is a foreign key referencing the DEPARTMENT relation.

- This means that a value of Dno in any tuple t_1 of the EMPLOYEE relation "**must match**" a value of the primary key of DEPARTMENT with the Dnumber attribute in some tuple t_2 of the DEPARTMENT relation or Dno can be NULL if the employee does not belong to a department.

Foreign key

Primary key

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

Referential integrity (or foreign key) constraint

- So,

statement of the constraint is as follows:

- The value in the foreign key column (or columns) FK of the **referencing relation** R_1 can be **either**:
  - (1) a value of an existing primary key that is a "corresponding primary key PK" in the **referenced relation** R_2,

    or

  - (2) a **NULL**.

- In case (2), the FK in R_1 should **not** be a part of its own primary key.

Displaying a relational database schema and its constraints

- Each relation schema can be displayed as a row of attribute names (see next slide for an example).
- The name of the relation is written above the attribute names.
- The primary key attribute (or attributes) are underlined.
- A foreign key (referential integrity) constraints is displayed as a directed arc (arrow) from the foreign key attributes to the referenced table.
  - Can also point the primary key of the referenced relation for clarity.

Entities:

1- Product : ID, Name, Quantity

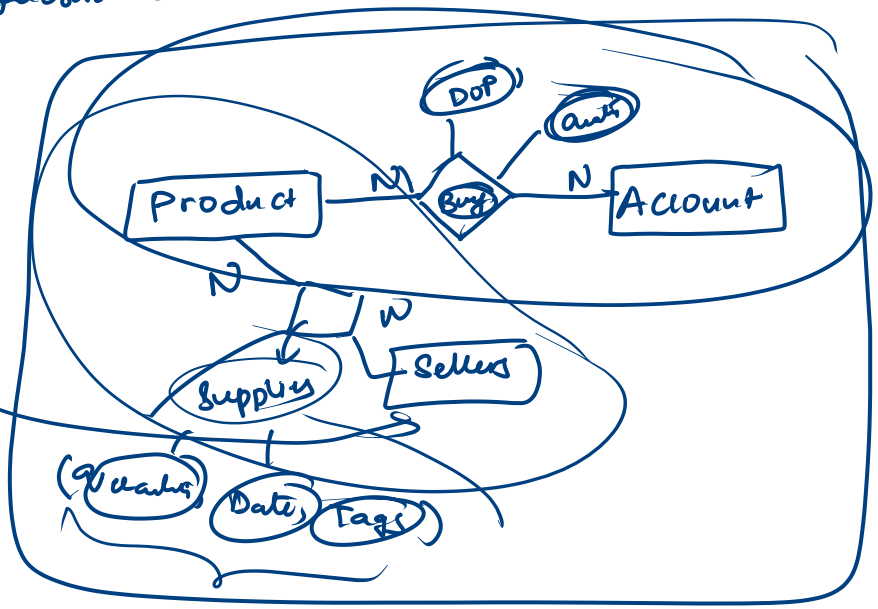
2. Account :- ID, Email, Address book

3. Sellers → (ID, Name, Add, Phone)

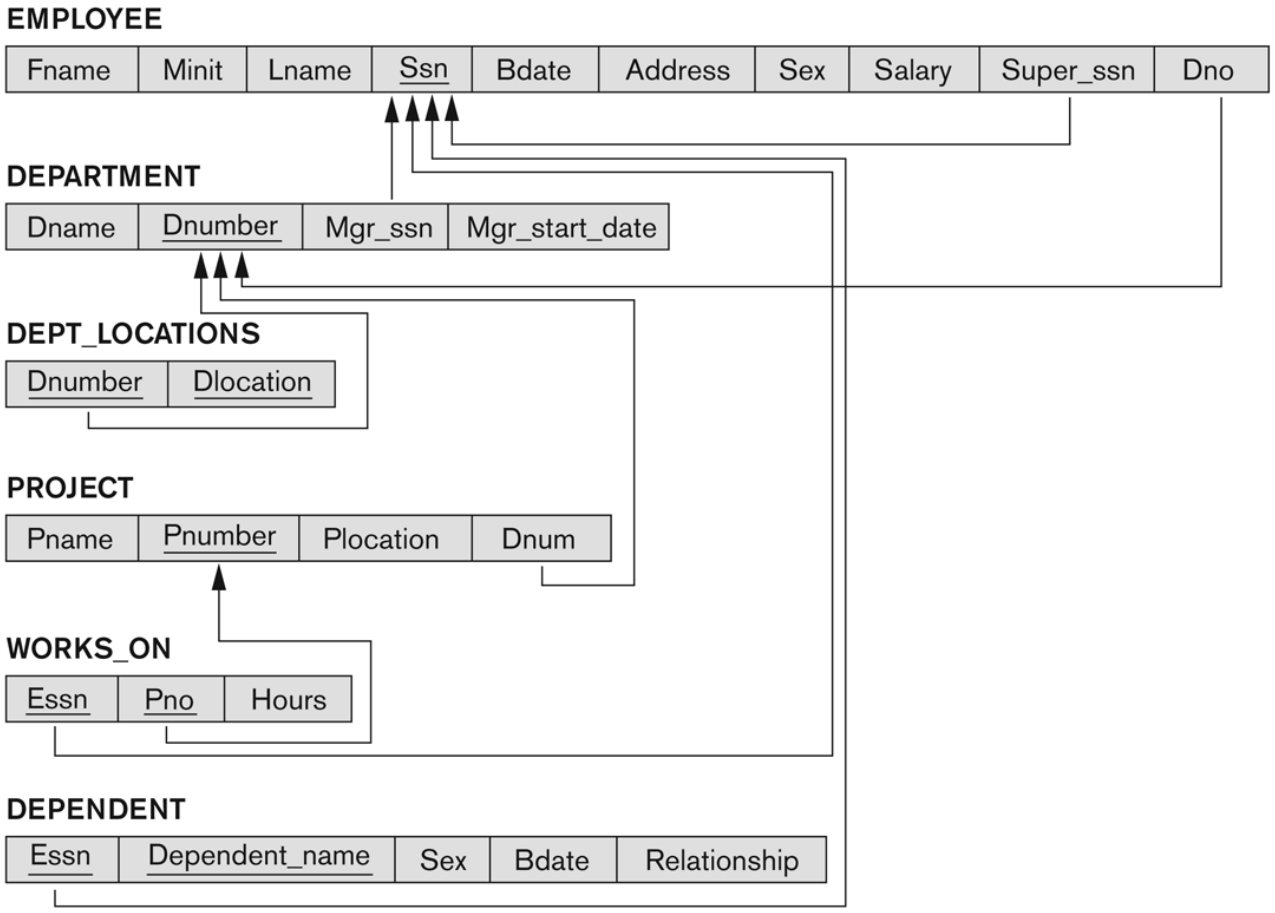
4. Employees

5. Departments

6. Projects



A referential integrity constraint can be displayed in a relational database schema as a directed arc (see below) from R_1.FK to R_2 (the relation it references).



Referential Integrity Constraints for COMPANY database.

Other types of constraints

- Semantic integrity constraints are based on application semantics and cannot be expressed by the DDL.
  - Example: "the max. no. of hours per employee for all projects he or she works on is 56 hours per week".
  - The salary of an employee should not exceed the salary of the employee's supervisor.
- A **constraint specification language** may have to be used to express these or can be specified and enforced within the **application programs**.
- Such constraints can also be specified in SQL that allows **triggers** and **assertions** through CREATE TRIGGER and CREATE ASSERTIONS.

Other types of constraints

- **State (static)** constraints – define the constraints that a valid state of the DB must satisfy.
- **Transition (dynamic)** constraints – to deal with state changes in the DB. Example, "the salary of an employee can only increase".
- They can be enforced by the application programs, or specified using active rules and triggers.

Populated DB state and Update Operations

- Each *relation* will have many tuples in its current relation state.
- The *relational database state* is a union of all the individual relation states.
- Whenever the database is changed, a new state arises.
- Basic operations for changing the database:
  - INSERT a new tuple in a relation.
  - DELETE an existing tuple from a relation.
  - MODIFY an attribute of an existing tuple.

INSERT operation

- It provides a list of attribute values for a new tuple *t* that is to be inserted into a relation *R*.

- It can violate any of the four constraints:

1. Domain constraints
2. Key constraints
3. Entity integrity
4. Referential integrity

Possible violations for 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.
  - Referential integrity:
    - if a foreign key value in the new tuple references a primary key value that does not exist in the referenced relation.
  - Entity integrity:
    - if the primary key value is NULL in the new tuple.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

One possible DB state for the COMPANY relational DB schema.

Example (see previous slide for DB)

- *Operation*:
  Insert <'Cecilia', 'F', 'Kolonsky', NULL, '1960-04-05', '6357 Windy Lane, Katy, TX', F, 28000, NULL, 4> into EMPLOYEE.
  Result: This insertion violates the entity integrity constraint (NULL for the primary key Ssn), so it is rejected. ✗
- *Operation*:
  Insert <'Alicia', 'J', 'Zelaya', '999887777', '1960-04-05', '6357 Windy Lane, Katy, TX', F, 28000, '987654321', 4> into EMPLOYEE.
  Result: This insertion violates the key constraint because another tuple with the same Ssn value already exists in the EMPLOYEE relation, and so it is rejected. ✗
- *Operation*:
  Insert <'Cecilia', 'F', 'Kolonsky', '677678989', '1960-04-05', '6357 Windswept, Katy, TX', F, 28000, '987654321', 7> into EMPLOYEE.
  Result: This insertion violates the referential integrity constraint specified on Dno in EMPLOYEE because no corresponding referenced tuple exists in DEPARTMENT with Dnumber = 7. ✗
- *Operation*:
  Insert <'Cecilia', 'F', 'Kolonsky', '677678989', '1960-04-05', '6357 Windy Lane, Katy, TX', F, 28000, NULL, 4> into EMPLOYEE.
  Result: This insertion satisfies all constraints, so it is acceptable. ✓

Actions for INSERT violations

- Default: reject the insertion.
- Attempt to correct the reason for rejecting the insertion.
- Ask the user to specify a correct/valid value or provide suggestions.
- Execute a user-specified error-correction routine.
- Trigger additional updates so that the violation is corrected (CASCADE option, SET NULL option).

DELETE operation

- It can violate only referential integrity. This occurs if the tuple being deleted is referenced by foreign keys from other tuples in the DB.
- To specify the deletion, "a condition on the attributes of the relation" selects the tuple(s) to be deleted.
- Example: (see next slide for DB)
  - *Operation*:
    Delete the WORKS_ON tuple with Essn = '999887777' and Pno = 10.
    Result: This deletion is acceptable and deletes exactly one tuple. 
  - *Operation*:
    Delete the EMPLOYEE tuple with Ssn = '999887777'.
    Result: This deletion is not acceptable, because there are tuples in WORKS_ON that refer to this tuple. Hence, if the tuple in EMPLOYEE is deleted, referential integrity violations will result. 
  - *Operation*:
    Delete the EMPLOYEE tuple with Ssn = '333445555'.
    Result: This deletion will result in even worse referential integrity violations, because the tuple involved is referenced by tuples from the EMPLOYEE, DEPARTMENT, WORKS_ON, and DEPENDENT relations. 

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

One possible DB state for the COMPANY relational DB schema.

Actions for DELETE violations

- Can be remedied by several actions:
  - RESTRICT option: reject the deletion.
  - CASCADE option: attempt to cascade (or propagate) by deleting 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.

Quiz

If a referencing attribute that causes a violation is part of the primary key, it cannot be set to NULL, otherwise it would violate entity integrity.

UPDATE operation

- It is used to change the values of one or more attributes in a tuple(s) of some relation *R* "through a condition on the attributes" of the relation to select the tuple(s) to be modified.

- Example: (see next slide for DB)
  - *Operation*:
    Update the salary of the EMPLOYEE tuple with Ssn = '999887777' to 28000.
    Result: Acceptable. ✓
  - *Operation*:
    Update the Dno of the EMPLOYEE tuple with Ssn = '999887777' to 1.
    Result: Acceptable. ✓
  - *Operation*:
    Update the Dno of the EMPLOYEE tuple with Ssn = '999887777' to 7.
    Result: Unacceptable, because it violates referential integrity. ✗
  - *Operation*:
    Update the Ssn of the EMPLOYEE tuple with Ssn = '999887777' to '987654321'.
    Result: Unacceptable, because it violates primary key constraint by repeating a value that already exists as a primary key in another tuple; it violates referential integrity constraints because there are other relations that refer to the existing value of Ssn. ✗

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

One possible DB state for the COMPANY relational DB schema.

Actions for UPDATE violations

- Updating an attribute that is neither part of a primary key nor part of a foreign key causes no problem. ✓

- So, UPDATE may violate domain constraint and NOT NULL constraint on an attribute being modified.
  - Updating an ordinary attribute (neither PK nor FK) causes no problems. ✓
    - It can only violate domain constraints.

- Any of the other constraints may also be violated, depending on the attribute being updated:
  - Updating the primary key (PK):
    - Handle similar to a DELETE followed by an INSERT.
  - Updating a foreign key (FK):
    - May violate referential integrity. DBMS must make sure that the new value refers to an existing tuple in the referenced relation (or is set to NULL).

Transaction

- A transaction is an executing program that includes some DB operations such as reading, insertion, deletion, updates to the DB.

- At the end of the transaction, it must leave the DB in a valid state.

- A single transaction may involve both retrievals and updates forming an "atomic unit".

  - Example, a transaction to apply bank withdrawal, and update the record by the withdrawal amount.

Summary

- Presented relational model concepts:
  - Definitions
  - Characteristics of relations
- Discussed relational model constraints and relational DB schemas:
  - Domain constraints
  - Key constraints
  - Entity integrity
  - Referential integrity
- Described the relational update operations (INSERT, DELETE, UPDATE) and dealt with the constraint violations.

Practice questions (check yourself)

1. What are the three main categories of constraints?
2. Explain schema-based constraints.
3. What is superkey? In this context, explain uniqueness constraint.
4. What is key constraint?
5. Discuss entity integrity constraints and referential integrity constraints.
6. What type of constraints can be violated by Insert operation?
7. What is a transaction?