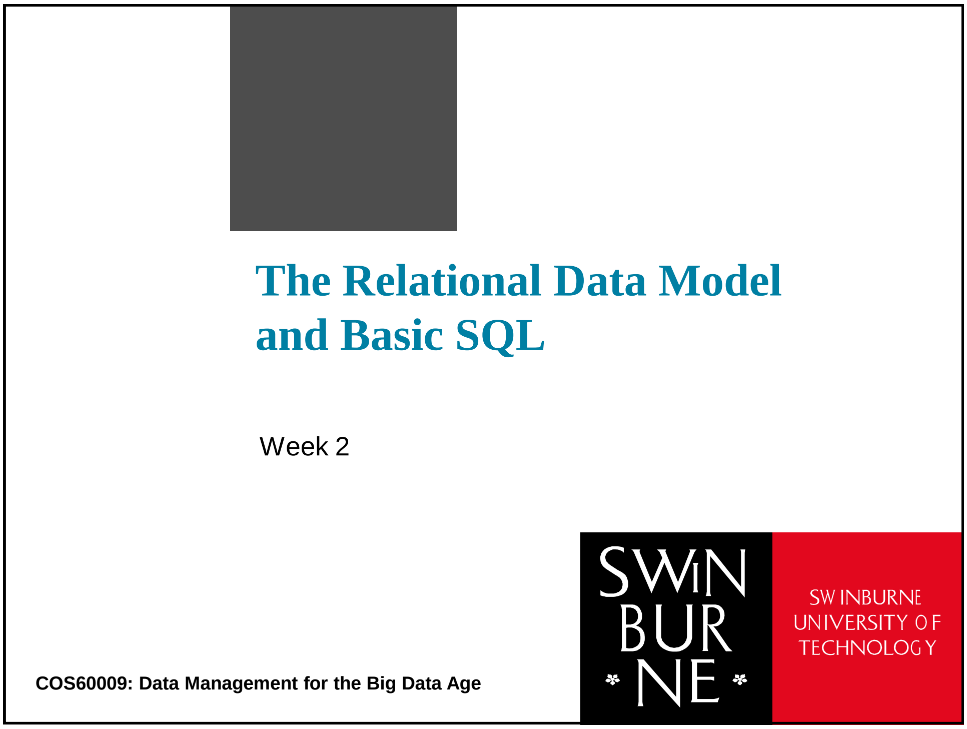


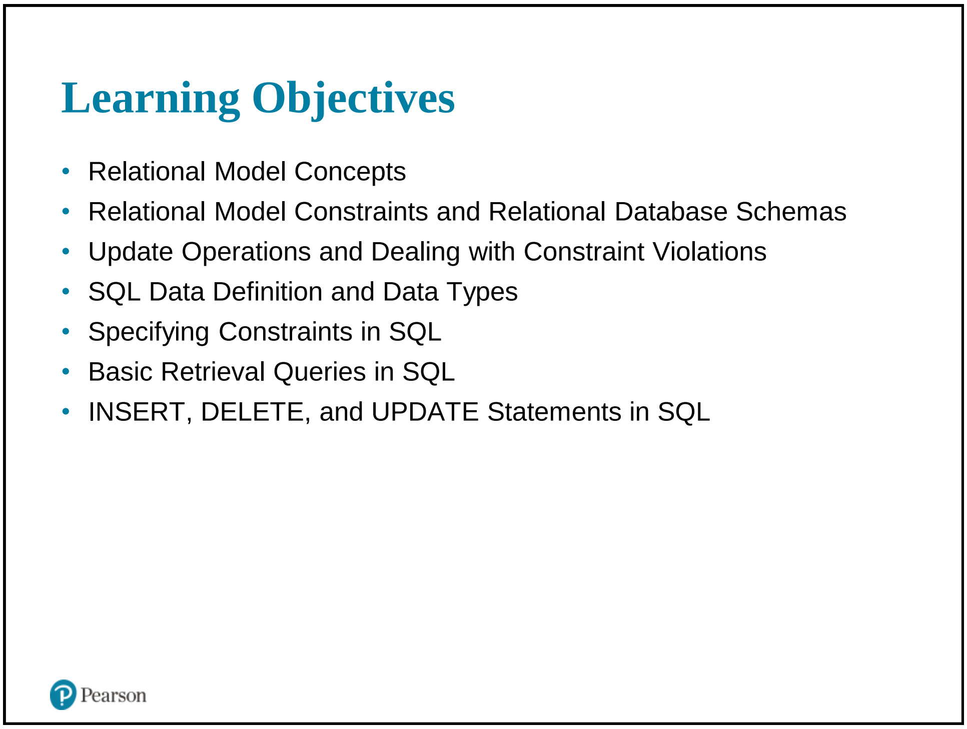
The Relational Data Model and Basic SQL

Week 2

COS60009: Data Management for the Big Data Age

SWINBURNE UNIVERSITY OF TECHNOLOGY

1



Learning Objectives

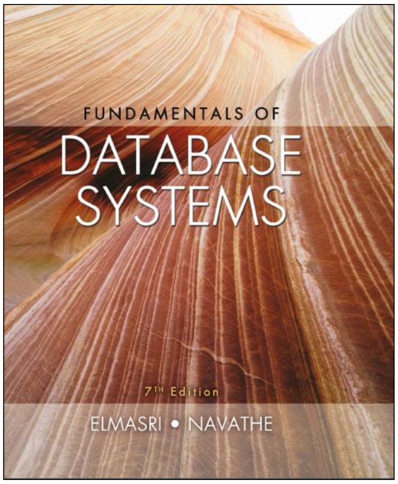
- Relational Model Concepts
- Relational Model Constraints and Relational Database Schemas
- Update Operations and Dealing with Constraint Violations
- SQL Data Definition and Data Types
- Specifying Constraints in SQL
- Basic Retrieval Queries in SQL
- INSERT, DELETE, and UPDATE Statements in SQL

Pearson

2

Fundamentals of Database Systems

Seventh Edition



Chapter 5

The Relational Data Model
and Relational Database
Constraints

Chapter 6

Basic SQL

3

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 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

4

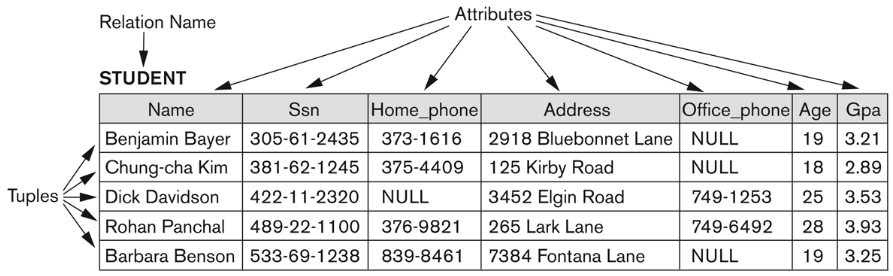
Informal Definitions

- Informally, a **relation** looks like a **table,** contains a **set of rows.**
- Each **row** corresponds to a real-world **entity** or **relationship**, has data elements for **columns**
  - 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**)
- Key of a Relation:
  - Each row has a value of a data item (or set of items) that uniquely identifies that row in the table
    - Called the **key**
  - In the STUDENT table, SSN is the key
  - Sometimes row-ids or sequential numbers are assigned as keys to identify the rows in a table
    - Called **artificial key** or **surrogate key**

Pearson

Example of a Relation

Figure 5.1 The attributes and tuples of a relation 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

Pearson

Formal Definitions - Schema

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

Pearson

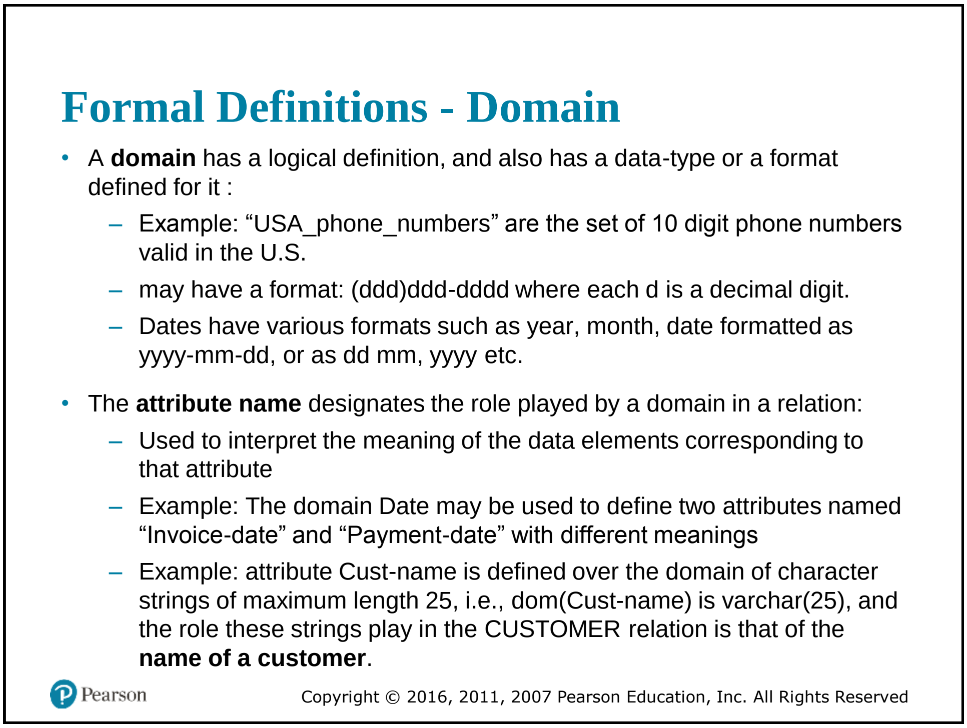
7

Formal Definitions - 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 is a 4-tuple and would consist of four values, for example:
  - <632895, "John Smith", "101 Main St. Atlanta, GA 30332", "(404) 894-2000">
  - This is called a 4-tuple as it has 4 values
- A relation is a **set** of such tuples (rows)

Pearson

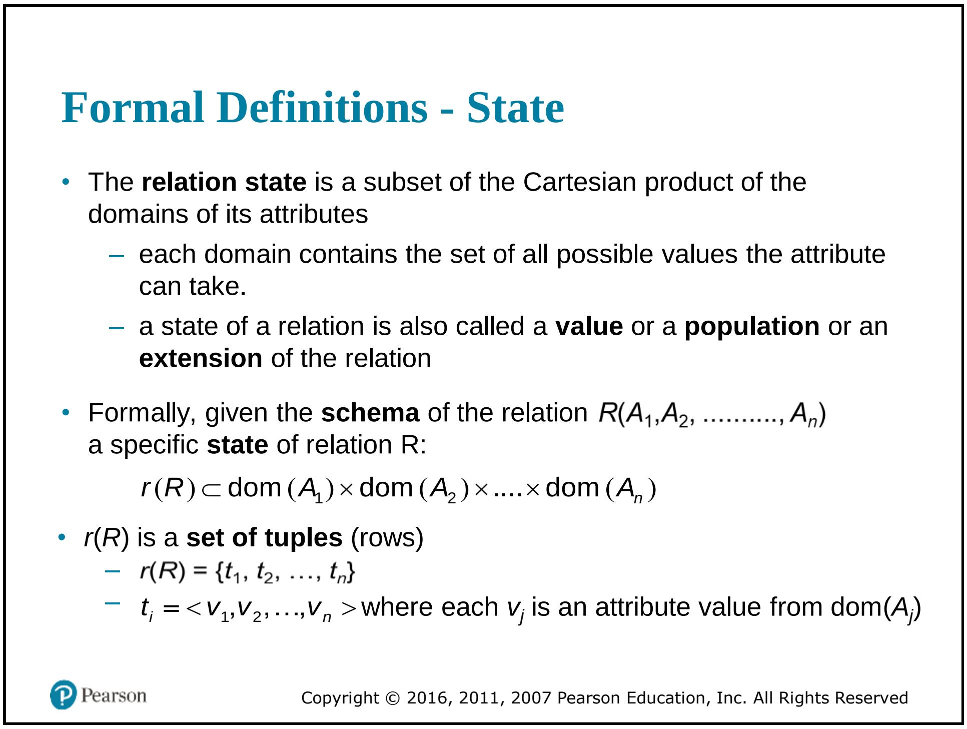
8



Formal Definitions - Domain

- A **domain** has a logical definition, and also has a data-type or a format defined for it :
  - Example: “USA_phone_numbers” are the set of 10 digit phone numbers valid in the U.S.
  - may have a format: (ddd)ddd-dddd where each d is a 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
  - Example: attribute Cust-name is defined over the domain of character strings of maximum length 25, i.e., dom(Cust-name) is varchar(25), and the role these strings play in the CUSTOMER relation is that of the **name of a customer**.

Pearson



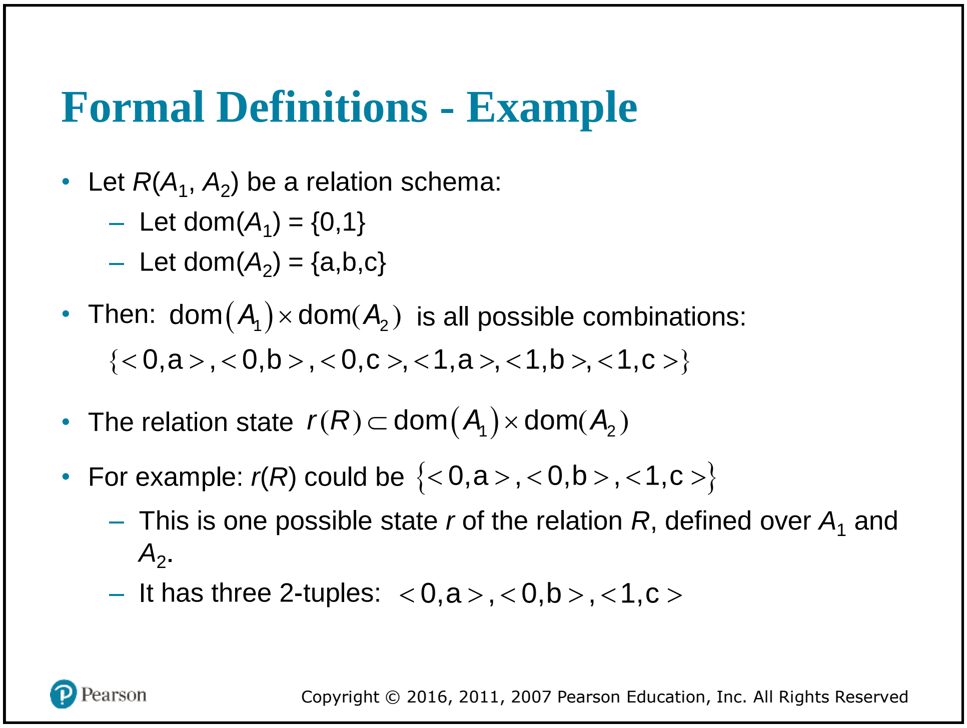
Formal Definitions - State

- 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.
  - a state of a relation is also called a **value** or a **population** or an **extension** of the relation
- Formally, given the **schema** of the relation $R(A_1, A_2, ..., A_n)$ a specific **state** of relation R:

  $$r(R) \subset \mathrm{dom}(A_1) \times \mathrm{dom}(A_2) \times .... \times \mathrm{dom}(A_n)$$

- $r(R)$ is a **set of tuples** (rows)
  - $r(R) = \{t_1, t_2, \ldots, t_n\}$
  - $t_i = < v_1, v_2, \ldots, v_n >$ where each v_j is an attribute value from dom(A_j)

Pearson

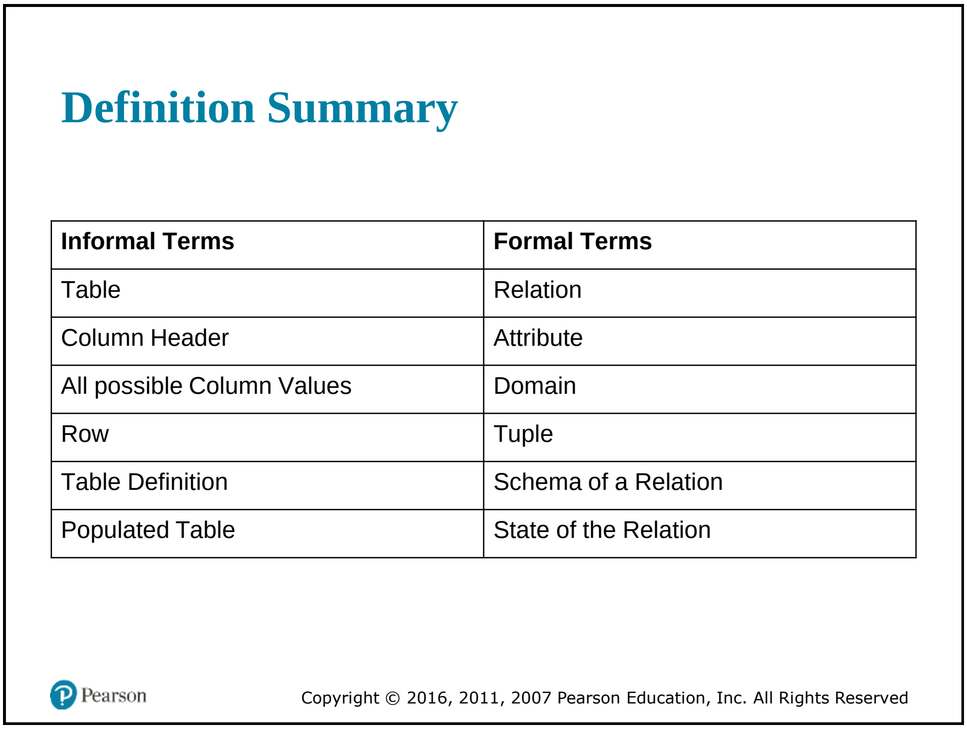


Formal Definitions - Example

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

  $\{<0,a>, <0,b>, <0,c>, <1,a>, <1,b>, <1,c>\}$
- The relation state $r(R) \subset \text{dom}(A_1) \times \text{dom}(A_2)$
- For example: $r(R)$ could be $\{<0,a>, <0,b>, <1,c>\}$
  - This is one possible state r of the relation R, defined over A_1 and A_2.
  - It has three 2-tuples: $<0,a>, <0,b>, <1,c>$

Pearson



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

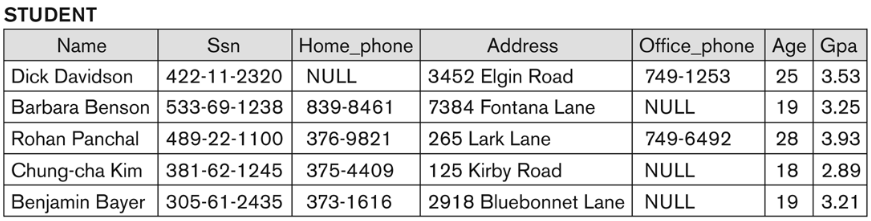
Pearson

Characteristics of Relations (1 of 2)

- 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.
- Ordering of attributes in a relation schema *R* (and of values within each tuple):
  - We will consider the attributes in $R(A_1, A_2, \ldots, A_n)$ and the values in $t = < v_1, v_2, \ldots, v_n >$ to be ordered.
    - (However, a more general alternative definition of relation does not require this ordering. It includes both the name and the value for each of the attributes.)

Same State as Previous Figure (but with Different Order of Tuples)

Figure 5.2 The relation STUDENT from Figure 5.1 with a different order of tuples.



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

Characteristics of Relations (2 of 2)

- 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 $t = < v_1, v_2, \ldots, v_n >$ is a tuple (row) in the relation state *r* of $R(A_1, A_2, \ldots, A_n)$
    - Then each v_i must be a value from **dom** (A_i)
  - A special **null** value is used to represent values that are unknown or not available or inapplicable in certain tuples.
- Notation:
  - We refer to **component values** of a tuple *t* by:
    - $t\,[A_i]$ or $t.A_i$
    - This is the value v_i of attribute A_i for tuple t
  - Similarly, $t\,[A_u, A_v, \ldots, A_w]$ refers to the subtuple of *t* containing the values of attributes $A_u, A_v, \ldots, A_w,$ respectively in *t*

Pearson

15

Constraints

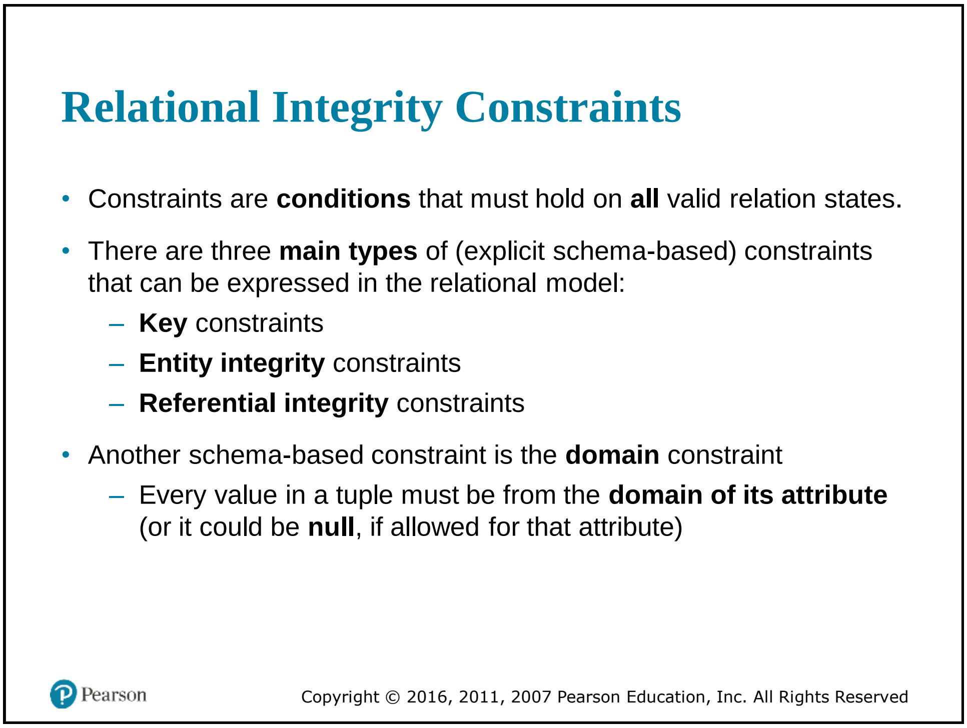
Constraints determine which values are permissible and which are not in the database.

They are of three main types:

1. **Inherent or Implicit Constraints:** These are based on the data model itself. (E.g., relational model does not allow a list as a value for any attribute)
2. **Schema-based or Explicit Constraints:** They are expressed in the schema by using the facilities provided by the model. (E.g., max. cardinality ratio constraint in the ER model)
3. **Application based or semantic constraints:** These are beyond the expressive power of the model and must be specified and enforced by the application programs. (E.g., the maximum number of hours per employee for all projects he or she works on is 56 hrs per week)

Pearson

16

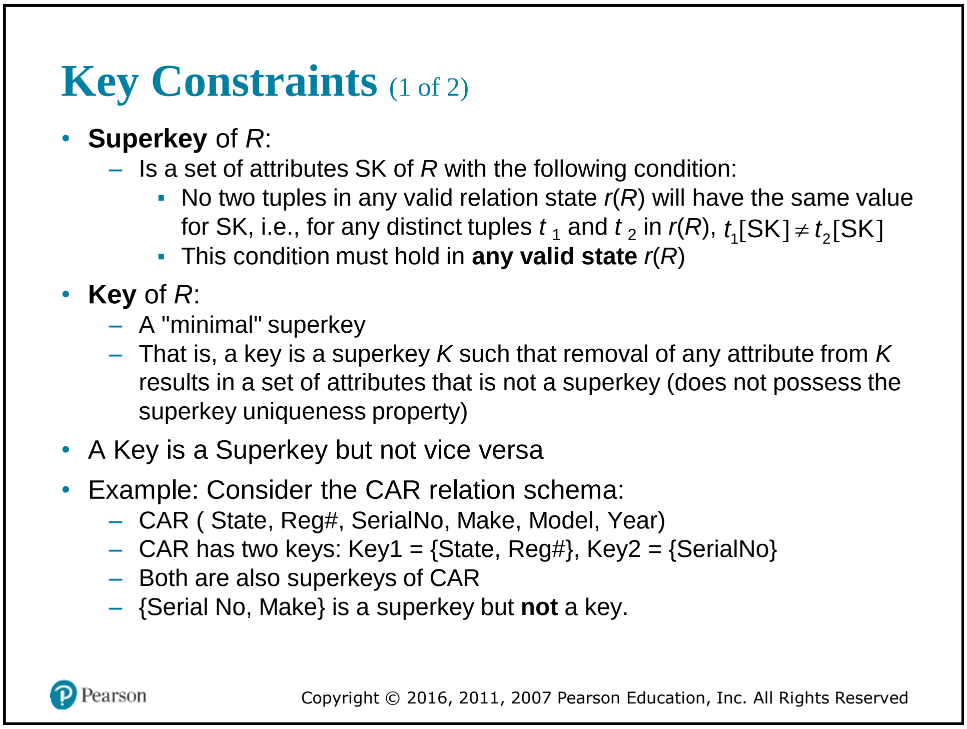


Relational Integrity Constraints

- Constraints are **conditions** that must hold on **all** valid relation states.
- There are three **main types** of (explicit schema-based) constraints that can be expressed in the relational model:
  - **Key** constraints
  - **Entity integrity** constraints
  - **Referential integrity** constraints
- Another schema-based 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)

Pearson

17



Key Constraints (1 of 2)

- **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 value for SK, i.e., for any distinct tuples t_1 and t_2 in *r*(*R*), $t_1[\text{SK}] \neq t_2[\text{SK}]$
    - 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 from *K* results in a set of attributes that is not a superkey (does not possess the superkey uniqueness property)
- A Key is a Superkey but not vice versa
- Example: Consider the CAR relation schema:
  - CAR (State, Reg#, SerialNo, Make, Model, Year)
  - CAR has two keys: Key1 = {State, Reg#}, Key2 = {SerialNo}
  - Both are also superkeys of CAR
  - {Serial No, Make} is a superkey but **not** a key.

Pearson

18

Key Constraints (2 of 2)

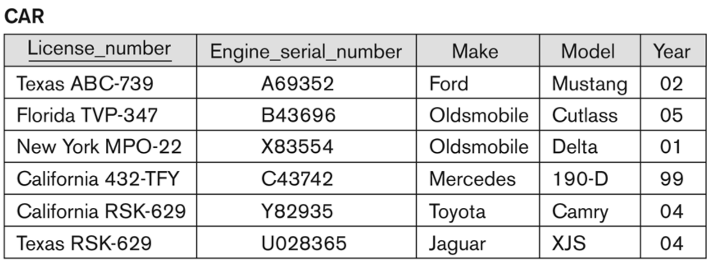
- If a relation has several **candidate keys,** one is chosen arbitrarily to be the **primary key.**
  - The primary key attributes are **underlined.**
- Example: Consider the CAR relation schema:
  - CAR (State, Reg#, SerialNo, Make, Model, Year)
  - We chose SerialNo as the primary key
- The primary key value is used to **uniquely identify** each tuple in a relation
  - Provides the tuple identity
- Also used to **reference** the tuple from another tuple
  - General rule: Choose as primary key the smallest of the candidate keys (in terms of size)
  - Not always applicable – choice is sometimes subjective

Pearson

19

Car Table with Two Candidate Keys – LicenseNumber Chosen as Primary Key

Figure 5.4 The CAR relation, with two candidate keys: License_number and Engine_serial_number.



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

Pearson

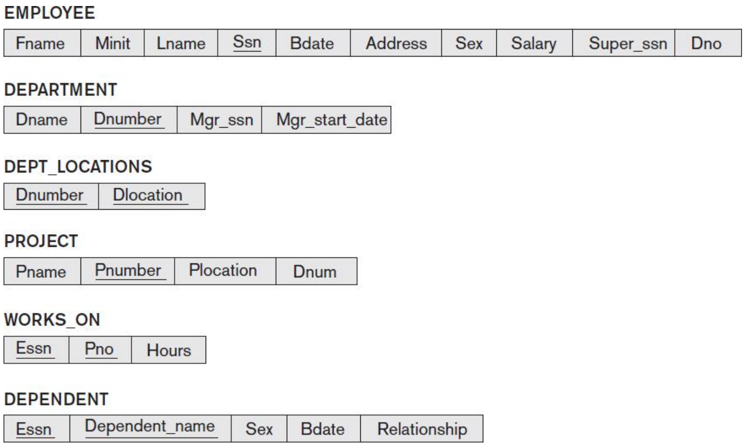
20

Relational Database Schema

- **Relational Database Schema:**
  - A set *S* of relation schemas that belong to the same database.
  - *S* is the name of the whole **database schema**
  - $S = \{R_1, R_2, ..., R_n\}$ and a set IC of integrity constraints.
  - $R_1, R_2, …, R_n$ are the names of the individual **relation schemas** within the database *S*
- Following slide shows a COMPANY database schema with 6 relation schemas

COMPANY Database Schema

Figure 5.5 Schema diagram for the COMPANY relational database schema.



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

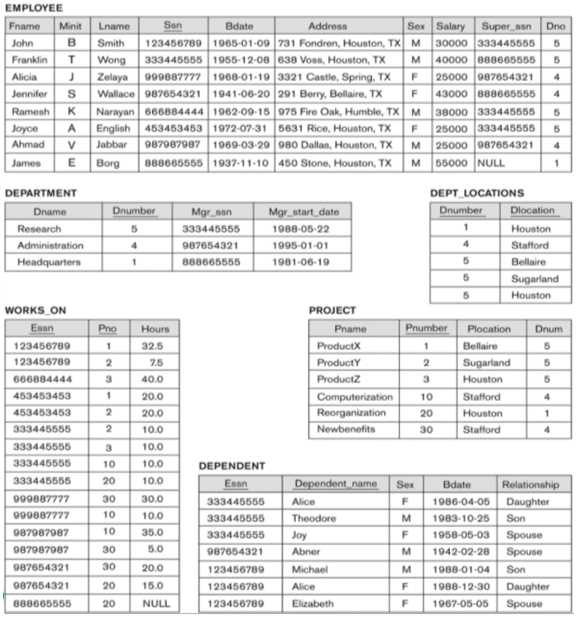
DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

Relational Database State

- 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 all the relation states satisfy the integrity constraints specified in IC.
- A relational database **state** is sometimes called a relational database **snapshot**.
- A database state that does not meet the constraints is an invalid state
- 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 state:
  - INSERT a new tuple in a relation
  - DELETE an existing tuple from a relation
  - MODIFY an attribute of an existing tuple

Populated Database State for COMPANY



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

Figure 5.6 One possible database state for the COMPANY relational database schema.

Entity Integrity

- The **primary key attributes** PK of each relation schema *R* in *S* cannot have null values in any tuple of *r*(*R*).
    - This is because primary key values are used to **identify** the individual tuples.
    - $t[\text{PK}] \neq \text{null}$ 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.

Pearson

25

Referential Integrity (or Foreign Key) Constraint

- 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.**
- Tuples in the **referencing relation** R_1 have attributes FK (called **foreign key** attributes) that reference the primary key attributes PK 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].
- A referential integrity constraint can be displayed in a relational database schema as a directed arc from R1.FK to R2
- Statement of the constraint
    - 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 value of 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.

Pearson

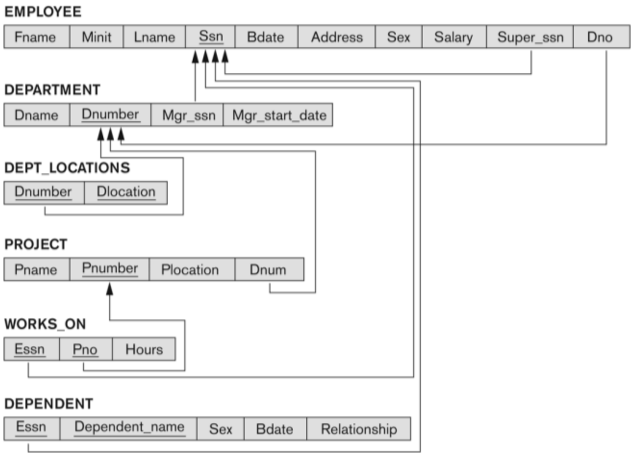
26

Displaying a Relational Database Schema and Its Constraints

- Each relation schema can be displayed as a row of attribute names
- The name of the relation is written above the attribute names
- The primary key attribute (or attributes) will be 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
- Next slide shows the COMPANY **relational schema diagram with referential integrity constraints**

Referential Integrity Constraints for COMPANY Database

Figure 5.7 Referential integrity constraints displayed on the COMPANY relational database schema.



EMPLOYEE

Fname	Minit	Lname	<u>Ssn</u>	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	<u>Dnumber</u>	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

<u>Dnumber</u>	<u>Dlocation</u>

PROJECT

Pname	<u>Pnumber</u>	Plocation	Dnum

WORKS_ON

<u>Essn</u>	<u>Pno</u>	Hours

DEPENDENT

<u>Essn</u>	<u>Dependent_name</u>	Sex	Bdate	Relationship

Update Operations on Relations

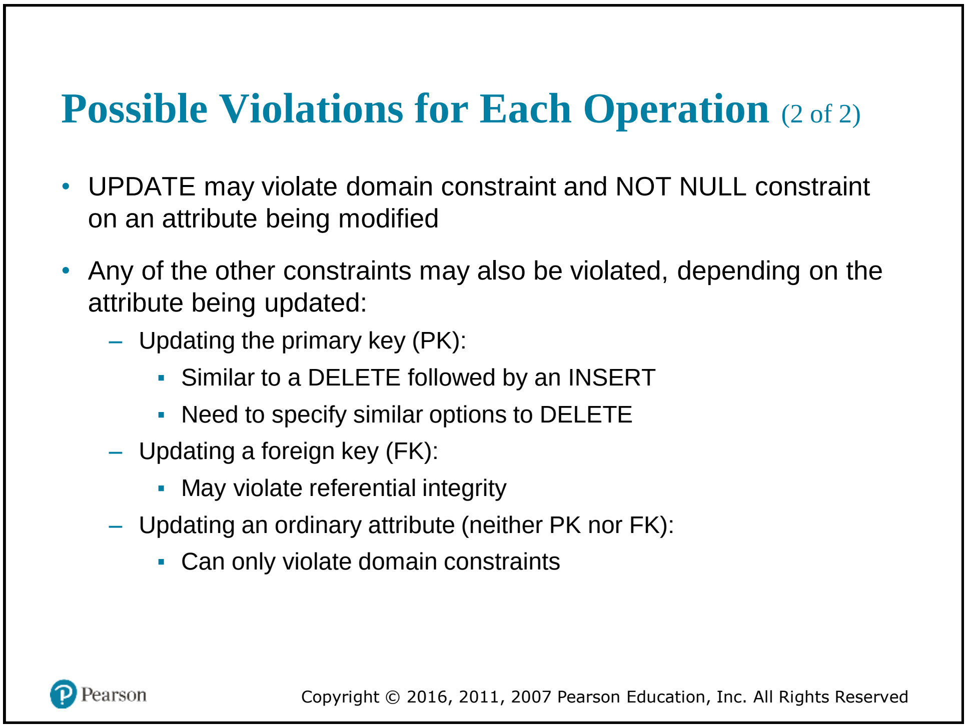
- INSERT a tuple.
- DELETE a tuple.
- MODIFY a tuple.
- Integrity constraints should not be violated by the update operations.
- Several update operations may have to be grouped together.
- Updates may **propagate** to cause other updates automatically. This may be necessary to maintain integrity constraints.
- In case of integrity violation, several actions can be taken:
  - Cancel the operation that causes the violation (RESTRICT or REJECT option)
  - Perform the operation but inform the user of the violation
  - Trigger additional updates so the violation is corrected (CASCADE option, SET NULL option)
  - Execute a user-specified error-correction routine

29

Possible Violations for Each Operation (1 of 2)

- 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
- 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: propagate the new primary key value into the foreign keys of the referencing tuples
      - 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

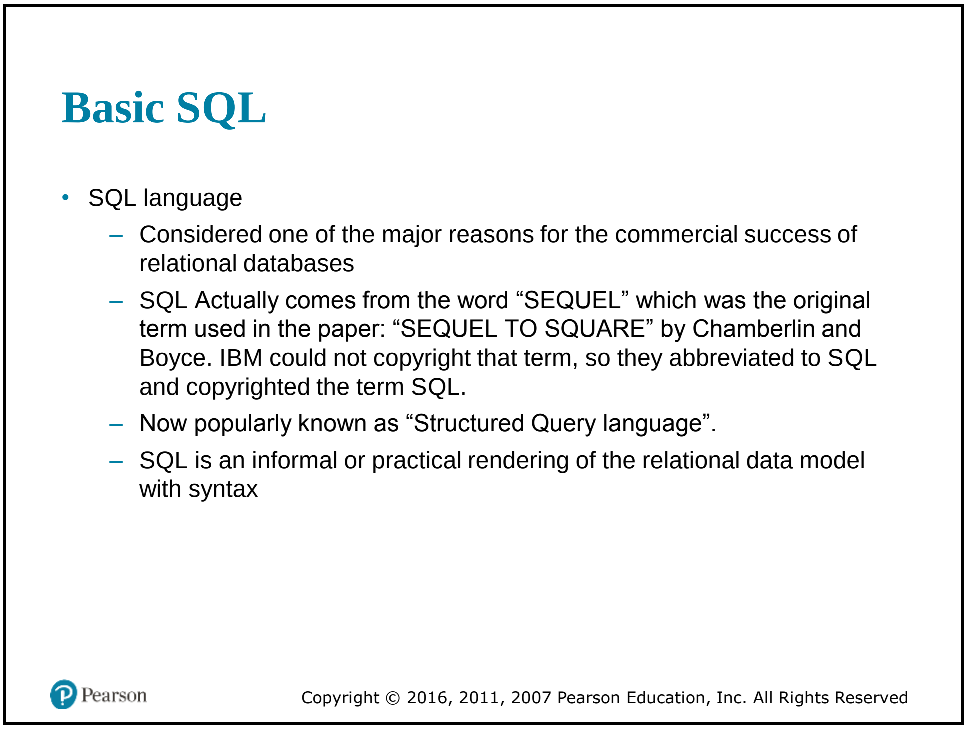
30



Possible Violations for Each Operation (2 of 2)

- UPDATE may violate domain constraint and NOT NULL constraint on an attribute being modified
- Any of the other constraints may also be violated, depending on the attribute being updated:
  - Updating the primary key (PK):
    - Similar to a DELETE followed by an INSERT
    - Need to specify similar options to DELETE
  - Updating a foreign key (FK):
    - May violate referential integrity
  - Updating an ordinary attribute (neither PK nor FK):
    - Can only violate domain constraints

31



Basic SQL

- SQL language
  - Considered one of the major reasons for the commercial success of relational databases
  - SQL Actually comes from the word “SEQUEL” which was the original term used in the paper: “SEQUEL TO SQUARE” by Chamberlin and Boyce. IBM could not copyright that term, so they abbreviated to SQL and copyrighted the term SQL.
  - Now popularly known as “Structured Query language”.
  - SQL is an informal or practical rendering of the relational data model with syntax

32

SQL Data Definition, Data Types, Standards

- Terminology:
  - **Table**, **row**, and **column** used for relational model terms relation, tuple, and attribute
- `CREATE` statement - Main SQL command for data definition
- The language has features for: Data definition, Data Manipulation, Transaction control, Indexing, Security specification (Grant and Revoke), Active databases, Multi-media, Distributed databases etc.
- SQL has gone through many standards: starting with SQL-86 or SQL 1.A. SQL-92 is referred to as SQL-2.
- Later standards (from SQL-1999) are divided into **core** specification and specialized **extensions.** The extensions are implemented for different applications – such as data mining, data warehousing, multimedia etc.
- SQL-2006 added XML features; In 2008 they added Object-oriented features.
- SQL-3 is the current standard which started with SQL-1999. It is not fully implemented in any RDBMS.

Pearson

33

Schema and Catalog Concepts in SQL

- **SQL schema**
  - Identified by a **schema name**
  - Includes an **authorization identifier** and **descriptors** for each element
- **Schema elements** include
  - Tables, constraints, views, domains, and other constructs
- Each statement in SQL ends with a **semicolon**
- `CREATE SCHEMA` statement
  - `CREATE SCHEMA COMPANY AUTHORIZATION 'Jsmith';`
- **Catalog**
  - Named collection of schemas in an SQL environment

Pearson

34

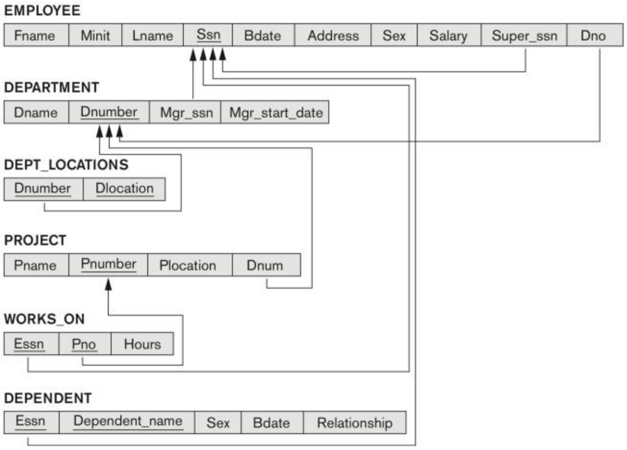
The CREATE TABLE Command in SQL

- Specifying a new relation
  - Provide name of table
  - Specify attributes, their types and initial constraints
- Can optionally specify schema:
  - `CREATE TABLE COMPANY.EMPLOYEE ...`

    or
  - `CREATE TABLE EMPLOYEE ...`
- **Base tables (base relations)**
  - Relation and its tuples are actually created and stored as a file by the DBMS
- **Virtual relations (views)**
  - Created through the `CREATE VIEW` statement. Do not correspond to any physical file.
- Some foreign keys may cause errors
  - Specified either via:
    - Circular references
    - Or because they refer to a table that has not yet been created

COMPANY Relational Database Schema

Figure 5.7 Referential integrity constraints displayed on the COMPANY relational database schema.



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

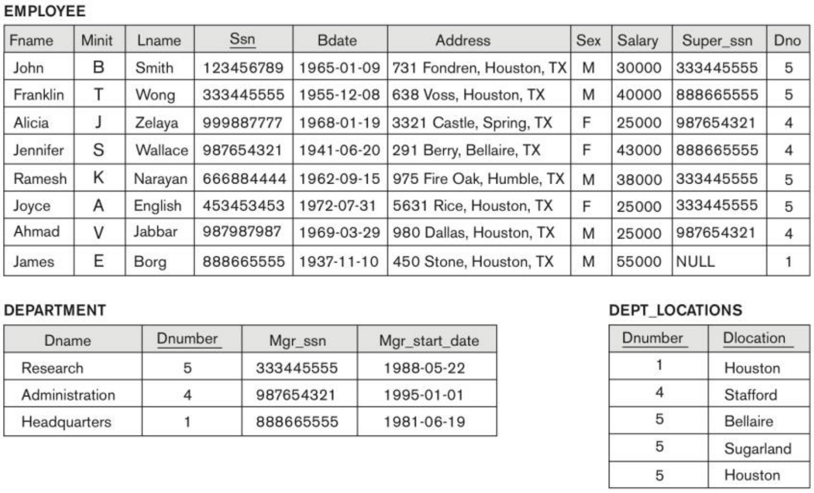
WORKS_ON

Essn	Pno	Hours

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

Figure 5.6 One Possible Database State for the COMPANY Relational Database Schema (1 of 2)



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

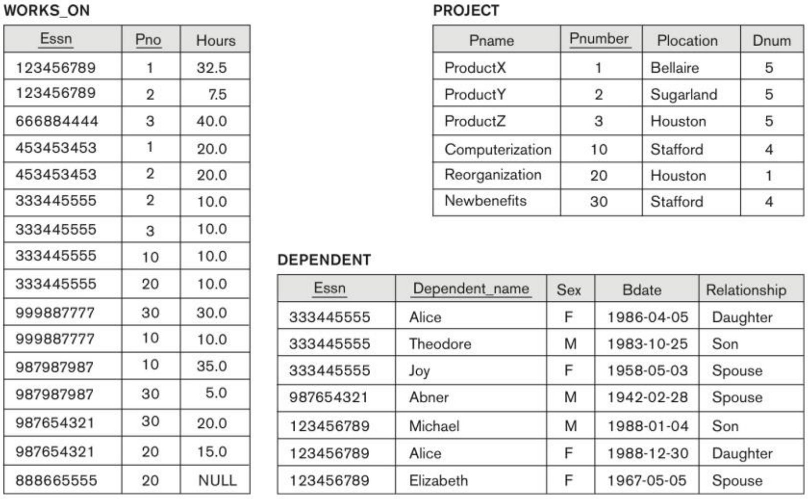
Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

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

Pearson

Figure 5.6 One Possible Database State for the COMPANY Relational Database Schema (2 of 2)



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

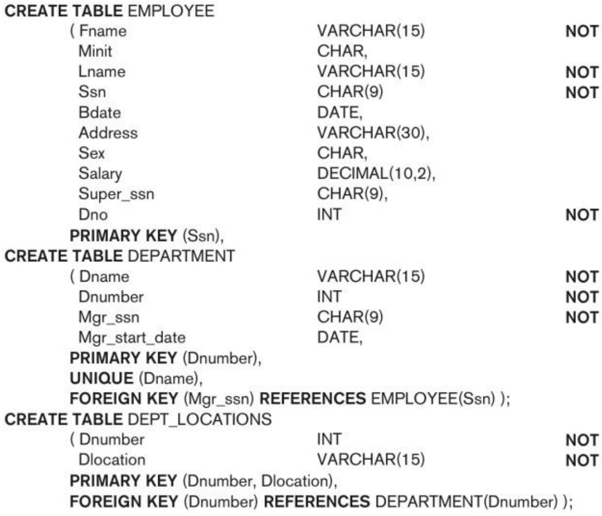
Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

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

Pearson

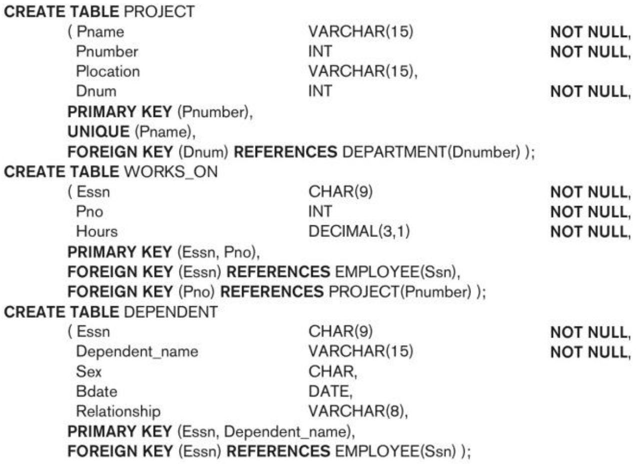
Figure 6.1 SQL CREATE TABLE Data Definition Statements for Defining the Company Schema from Figure 5.7 (1 of 2)



```
CREATE TABLE EMPLOYEE
        ( Fname                 VARCHAR(15)           NOT NULL,
          Minit                 CHAR,
          Lname                 VARCHAR(15)           NOT NULL,
          Ssn                   CHAR(9)               NOT NULL,
          Bdate                 DATE,
          Address               VARCHAR(30),
          Sex                   CHAR,
          Salary                DECIMAL(10,2),
          Super_ssn             CHAR(9),
          Dno                   INT                   NOT NULL,
        PRIMARY KEY (Ssn),
CREATE TABLE DEPARTMENT
        ( Dname                 VARCHAR(15)           NOT NULL,
          Dnumber               INT                   NOT NULL,
          Mgr_ssn               CHAR(9)               NOT NULL,
          Mgr_start_date        DATE,
        PRIMARY KEY (Dnumber),
        UNIQUE (Dname),
        FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn) );
CREATE TABLE DEPT_LOCATIONS
        ( Dnumber               INT                   NOT NULL,
          Dlocation             VARCHAR(15)           NOT NULL,
        PRIMARY KEY (Dnumber, Dlocation),
        FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber) );
```

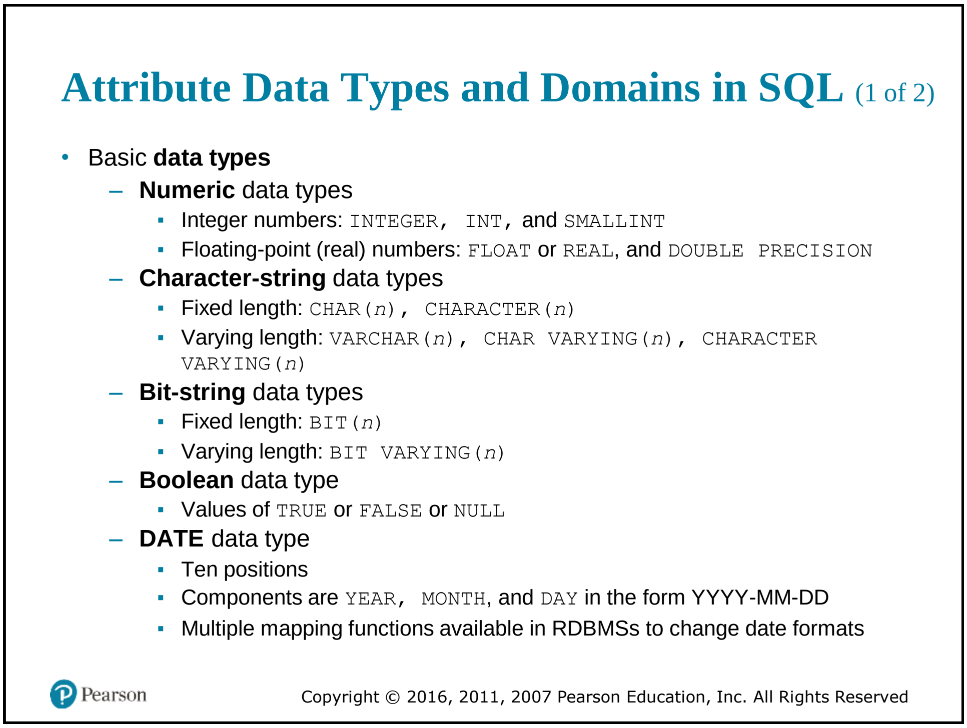
Pearson

Figure 6.1 SQL CREATE TABLE Data Definition Statements for Defining the Company Schema from Figure 5.7 (2 of 2)



```
CREATE TABLE PROJECT
        ( Pname                 VARCHAR(15)           NOT NULL,
          Pnumber               INT                   NOT NULL,
          Plocation             VARCHAR(15),
          Dnum                  INT                   NOT NULL,
        PRIMARY KEY (Pnumber),
        UNIQUE (Pname),
        FOREIGN KEY (Dnum) REFERENCES DEPARTMENT(Dnumber) );
CREATE TABLE WORKS_ON
        ( Essn                  CHAR(9)               NOT NULL,
          Pno                   INT                   NOT NULL,
          Hours                 DECIMAL(3,1)          NOT NULL,
        PRIMARY KEY (Essn, Pno),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn),
        FOREIGN KEY (Pno) REFERENCES PROJECT(Pnumber) );
CREATE TABLE DEPENDENT
        ( Essn                  CHAR(9)               NOT NULL,
          Dependent_name        VARCHAR(15)           NOT NULL,
          Sex                   CHAR,
          Bdate                 DATE,
          Relationship          VARCHAR(8),
        PRIMARY KEY (Essn, Dependent_name),
        FOREIGN KEY (Essn) REFERENCES EMPLOYEE(Ssn) );
```

Pearson

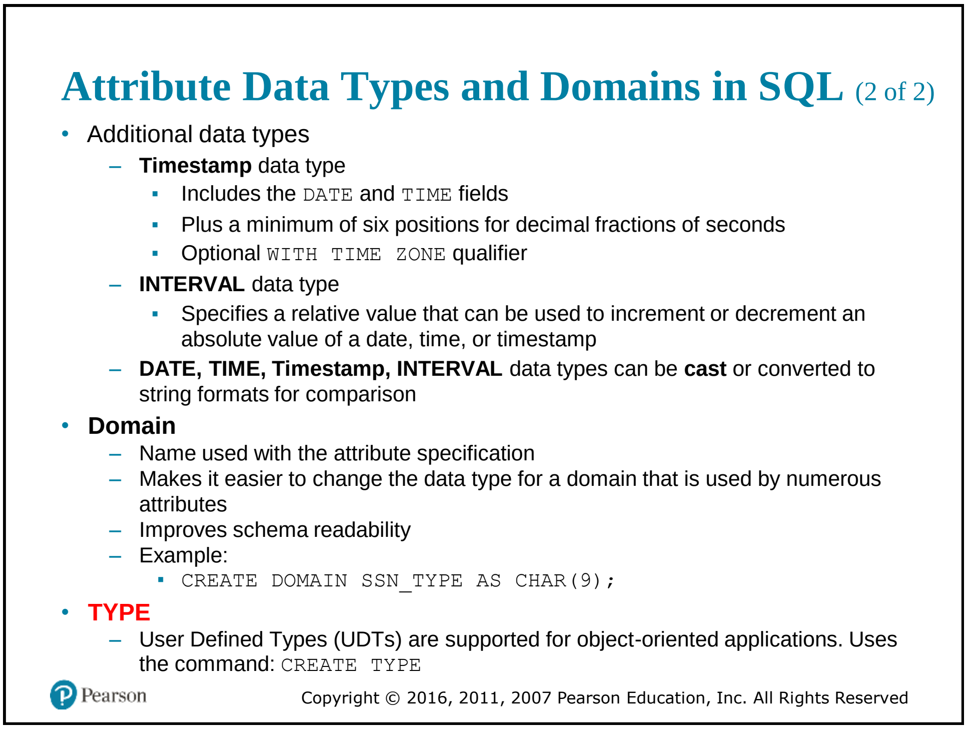


Attribute Data Types and Domains in SQL (1 of 2)

- Basic **data types**
  - **Numeric** data types
    - Integer numbers: `INTEGER`, `INT`, and `SMALLINT`
    - Floating-point (real) numbers: `FLOAT` or `REAL`, and `DOUBLE PRECISION`
  - **Character-string** data types
    - Fixed length: `CHAR(`*n*`)`, `CHARACTER(`*n*`)`
    - Varying length: `VARCHAR(`*n*`)`, `CHAR VARYING(`*n*`)`, `CHARACTER VARYING(`*n*`)`
  - **Bit-string** data types
    - Fixed length: `BIT(`*n*`)`
    - Varying length: `BIT VARYING(`*n*`)`
  - **Boolean** data type
    - Values of `TRUE` or `FALSE` or `NULL`
  - **DATE** data type
    - Ten positions
    - Components are `YEAR`, `MONTH`, and `DAY` in the form YYYY-MM-DD
    - Multiple mapping functions available in RDBMSs to change date formats

Pearson

41

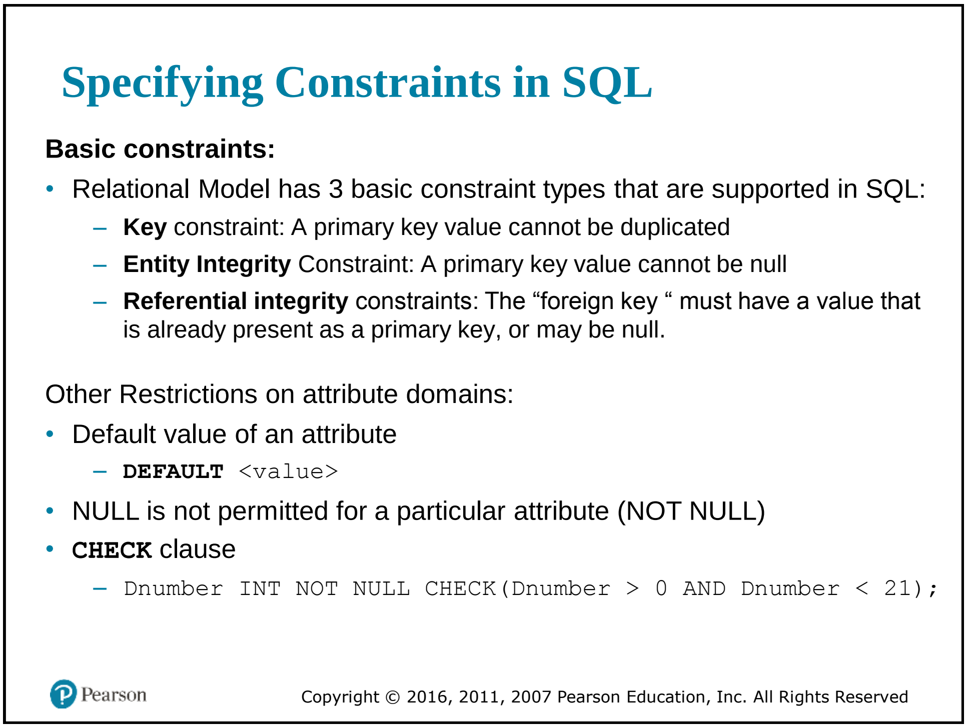


Attribute Data Types and Domains in SQL (2 of 2)

- Additional data types
  - **Timestamp** data type
    - Includes the `DATE` and `TIME` fields
    - Plus a minimum of six positions for decimal fractions of seconds
    - Optional `WITH TIME ZONE` qualifier
  - **INTERVAL** data type
    - Specifies a relative value that can be used to increment or decrement an absolute value of a date, time, or timestamp
  - **DATE, TIME, Timestamp, INTERVAL** data types can be **cast** or converted to string formats for comparison
- **Domain**
  - Name used with the attribute specification
  - Makes it easier to change the data type for a domain that is used by numerous attributes
  - Improves schema readability
  - Example:
    - `CREATE DOMAIN SSN_TYPE AS CHAR(9);`
- **TYPE**
  - User Defined Types (UDTs) are supported for object-oriented applications. Uses the command: `CREATE TYPE`

Pearson

42



Specifying Constraints in SQL

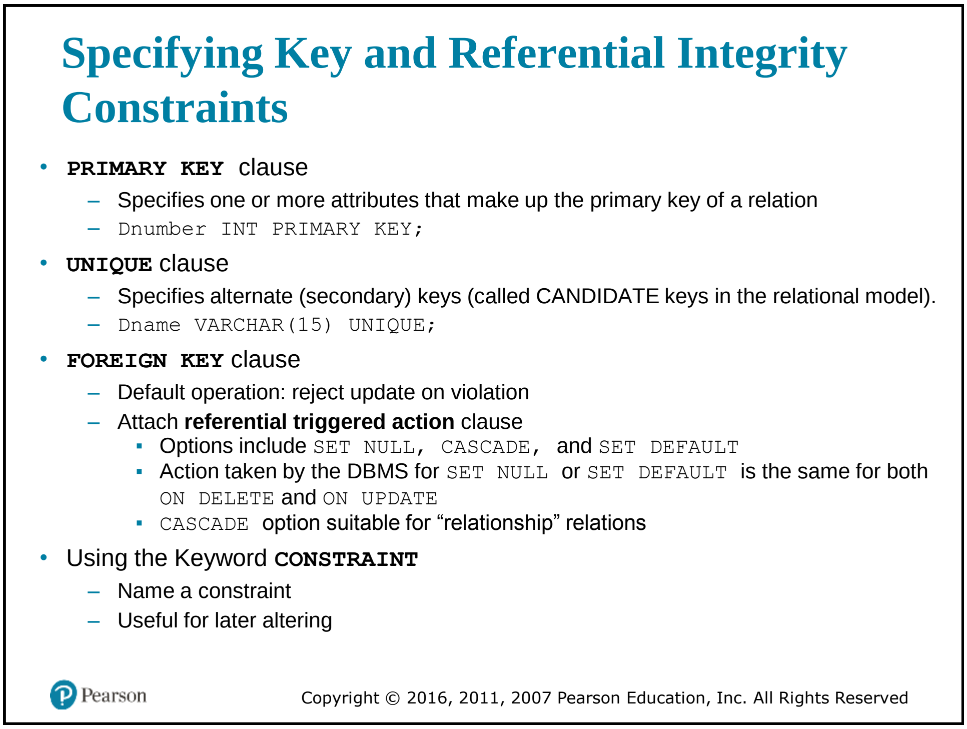
Basic constraints:

- Relational Model has 3 basic constraint types that are supported in SQL:
  - **Key** constraint: A primary key value cannot be duplicated
  - **Entity Integrity** Constraint: A primary key value cannot be null
  - **Referential integrity** constraints: The “foreign key “ must have a value that is already present as a primary key, or may be null.

Other Restrictions on attribute domains:

- Default value of an attribute
  - `DEFAULT <value>`
- NULL is not permitted for a particular attribute (NOT NULL)
- `CHECK` clause
  - `Dnumber INT NOT NULL CHECK(Dnumber > 0 AND Dnumber < 21);`

Pearson

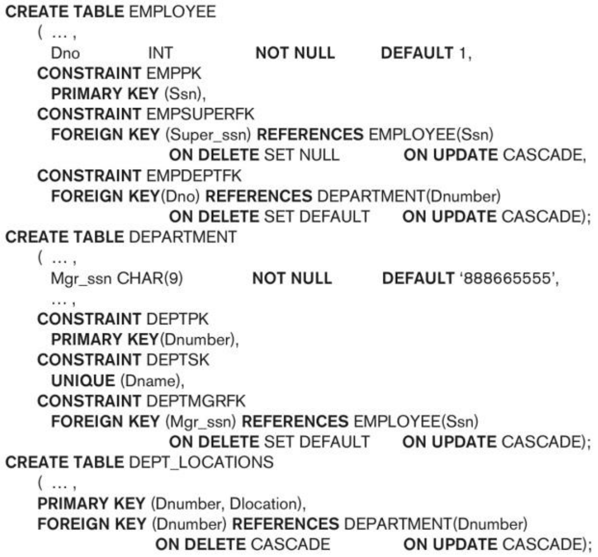


Specifying Key and Referential Integrity Constraints

- `PRIMARY KEY` clause
  - Specifies one or more attributes that make up the primary key of a relation
  - `Dnumber INT PRIMARY KEY;`
- `UNIQUE` clause
  - Specifies alternate (secondary) keys (called CANDIDATE keys in the relational model).
  - `Dname VARCHAR(15) UNIQUE;`
- `FOREIGN KEY` clause
  - Default operation: reject update on violation
  - Attach **referential triggered action** clause
    - Options include `SET NULL`, `CASCADE`, and `SET DEFAULT`
    - Action taken by the DBMS for `SET NULL` or `SET DEFAULT` is the same for both `ON DELETE` and `ON UPDATE`
    - `CASCADE` option suitable for “relationship” relations
- Using the Keyword `CONSTRAINT`
  - Name a constraint
  - Useful for later altering

Pearson

Figure 6.2 Default Attribute Values and Referential Integrity Triggered Action Specification



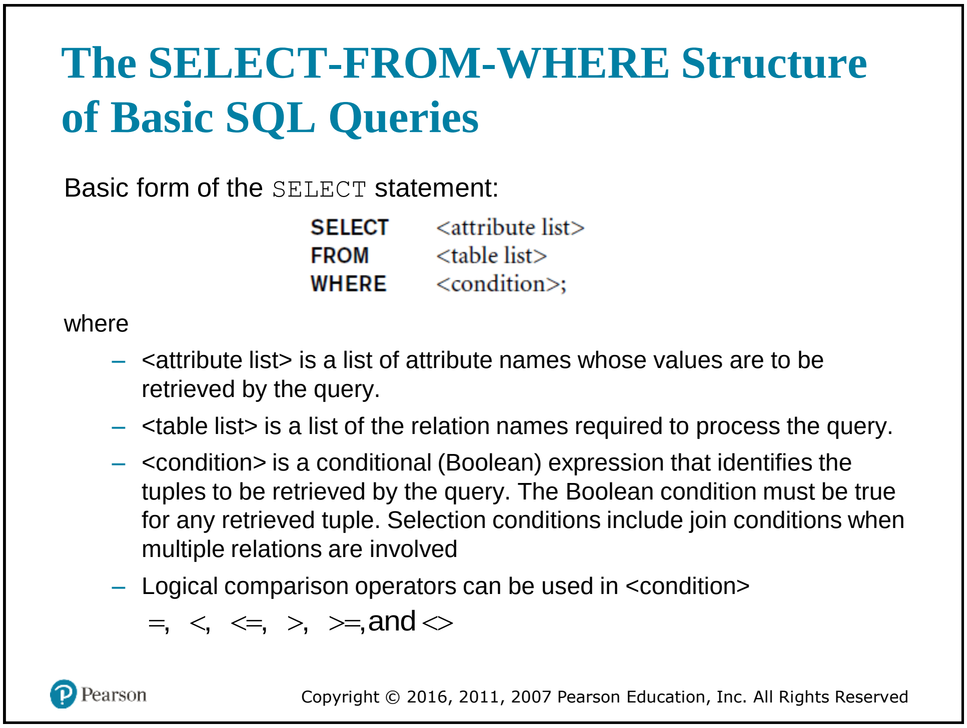
```
CREATE TABLE EMPLOYEE
    ( … ,
      Dno          INT           NOT NULL      DEFAULT 1,
    CONSTRAINT EMPPK
      PRIMARY KEY (Ssn),
    CONSTRAINT EMPSUPERFK
      FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn)
                    ON DELETE SET NULL           ON UPDATE CASCADE,
    CONSTRAINT EMPDEPTFK
      FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber)
                    ON DELETE SET DEFAULT        ON UPDATE CASCADE);
CREATE TABLE DEPARTMENT
    ( … ,
      Mgr_ssn CHAR(9)            NOT NULL      DEFAULT '888665555',
      … ,
    CONSTRAINT DEPTPK
      PRIMARY KEY(Dnumber),
    CONSTRAINT DEPTSK
      UNIQUE (Dname),
    CONSTRAINT DEPTMGRFK
      FOREIGN KEY (Mgr_ssn) REFERENCES EMPLOYEE(Ssn)
                    ON DELETE SET DEFAULT        ON UPDATE CASCADE);
CREATE TABLE DEPT_LOCATIONS
    ( … ,
    PRIMARY KEY (Dnumber, Dlocation),
    FOREIGN KEY (Dnumber) REFERENCES DEPARTMENT(Dnumber)
                  ON DELETE CASCADE              ON UPDATE CASCADE);
```

45

Basic Retrieval Queries in SQL

- `SELECT` statement
  - One basic statement for retrieving information from a database
- SQL allows a table to have two or more tuples that are identical in all their attribute values
  - Unlike relational model (relational model is strictly set-theory based)
  - Multiset or bag behavior
  - Tuple-id may be used as a key

46



The SELECT-FROM-WHERE Structure of Basic SQL Queries

Basic form of the SELECT statement:

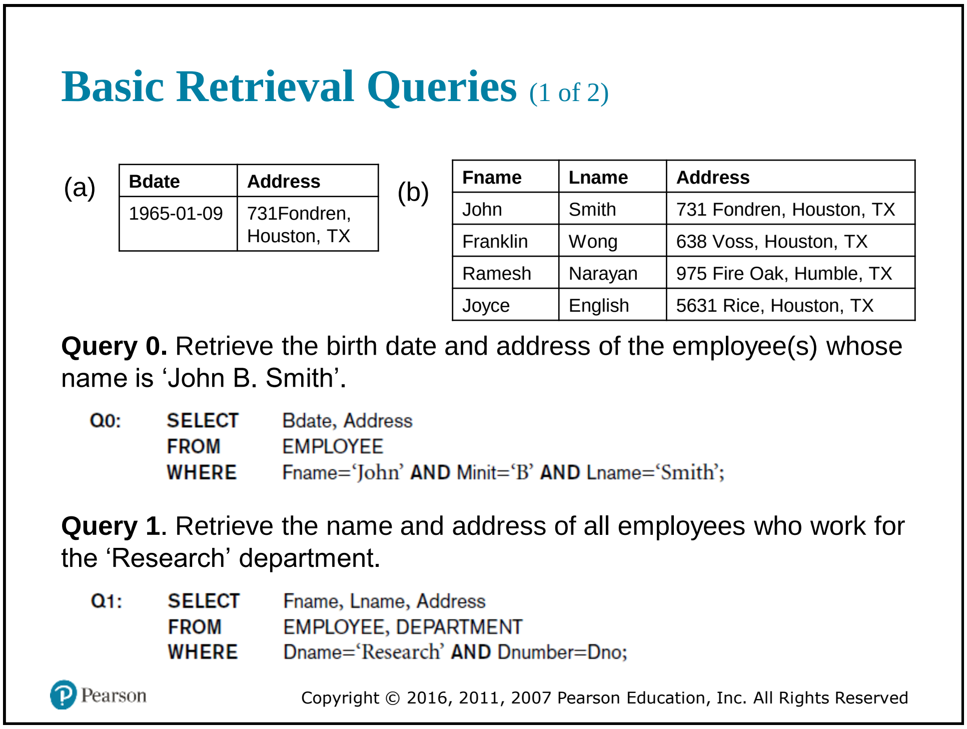
```
SELECT    <attribute list>
FROM      <table list>
WHERE     <condition>;
```

where

- <attribute list> is a list of attribute names whose values are to be retrieved by the query.
- <table list> is a list of the relation names required to process the query.
- <condition> is a conditional (Boolean) expression that identifies the tuples to be retrieved by the query. The Boolean condition must be true for any retrieved tuple. Selection conditions include join conditions when multiple relations are involved
- Logical comparison operators can be used in <condition>

  =, <, <=, >, >=, and <>

47



Basic Retrieval Queries (1 of 2)

(a)

Bdate	Address
1965-01-09	731Fondren, Houston, TX

(b)

Fname	Lname	Address
John	Smith	731 Fondren, Houston, TX
Franklin	Wong	638 Voss, Houston, TX
Ramesh	Narayan	975 Fire Oak, Humble, TX
Joyce	English	5631 Rice, Houston, TX

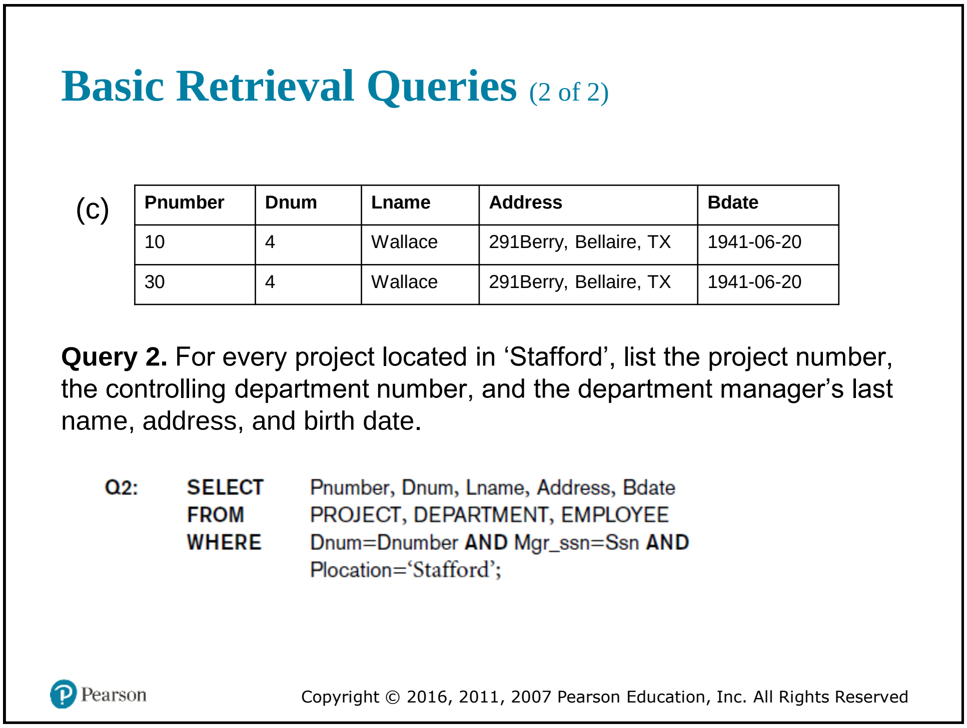
Query 0. Retrieve the birth date and address of the employee(s) whose name is ‘John B. Smith’.

```
Q0:   SELECT    Bdate, Address
      FROM      EMPLOYEE
      WHERE     Fname=‘John’ AND Minit=‘B’ AND Lname=‘Smith’;
```

Query 1. Retrieve the name and address of all employees who work for the ‘Research’ department.

```
Q1:   SELECT    Fname, Lname, Address
      FROM      EMPLOYEE, DEPARTMENT
      WHERE     Dname=‘Research’ AND Dnumber=Dno;
```

48



Basic Retrieval Queries (2 of 2)

(c)

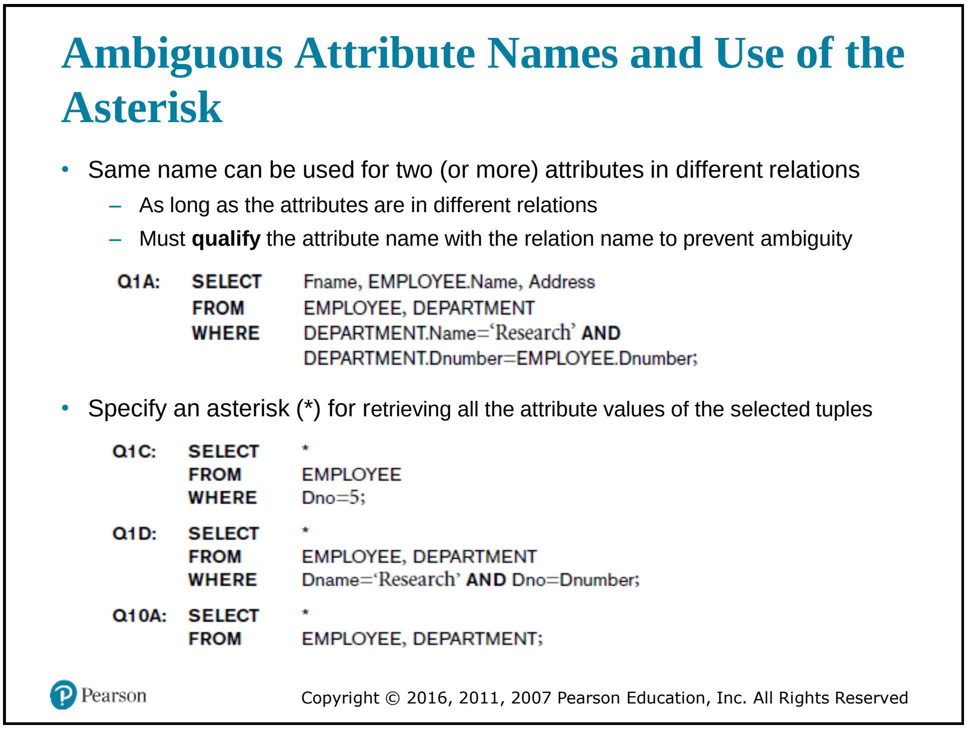
Pnumber	Dnum	Lname	Address	Bdate
10	4	Wallace	291Berry, Bellaire, TX	1941-06-20
30	4	Wallace	291Berry, Bellaire, TX	1941-06-20

Query 2. For every project located in ‘Stafford’, list the project number, the controlling department number, and the department manager’s last name, address, and birth date.

```
Q2:    SELECT   Pnumber, Dnum, Lname, Address, Bdate
       FROM     PROJECT, DEPARTMENT, EMPLOYEE
       WHERE    Dnum=Dnumber AND Mgr_ssn=Ssn AND
                Plocation=‘Stafford’;
```

Pearson

49



Ambiguous Attribute Names and Use of the Asterisk

- Same name can be used for two (or more) attributes in different relations
  - As long as the attributes are in different relations
  - Must **qualify** the attribute name with the relation name to prevent ambiguity

```
Q1A:   SELECT   Fname, EMPLOYEE.Name, Address
       FROM     EMPLOYEE, DEPARTMENT
       WHERE    DEPARTMENT.Name=‘Research’ AND
                DEPARTMENT.Dnumber=EMPLOYEE.Dnumber;
```

- Specify an asterisk (*) for retrieving all the attribute values of the selected tuples

```
Q1C:   SELECT   *
       FROM     EMPLOYEE
       WHERE    Dno=5;

Q1D:   SELECT   *
       FROM     EMPLOYEE, DEPARTMENT
       WHERE    Dname=‘Research’ AND Dno=Dnumber;

Q10A:  SELECT   *
       FROM     EMPLOYEE, DEPARTMENT;
```

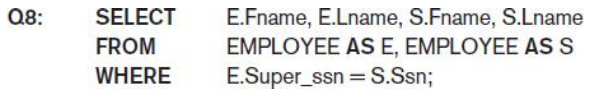
Pearson

50

Aliasing, Renaming and Tuple Variables

- **Aliases** or **tuple variables**
  - Declare alternative relation names E and S to refer to the EMPLOYEE relation twice in a query:

Query 8. For each employee, retrieve the employee's first and last name and the first and last name of his or her immediate supervisor.



```
Q8:   SELECT   E.Fname, E.Lname, S.Fname, S.Lname
      FROM     EMPLOYEE AS E, EMPLOYEE AS S
      WHERE    E.Super_ssn = S.Ssn;
```

  - Recommended practice to abbreviate names and to prefix same or similar attribute from multiple tables.
  - The attribute names can also be renamed

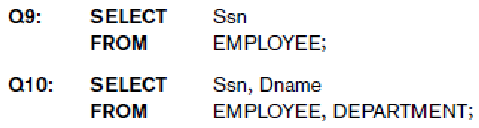
```
EMPLOYEE AS E(Fn, Mi, Ln, Ssn, Bd,
Addr, Sex, Sal, Sssn, Dno)
```

Note that the relation EMPLOYEE now has a variable name E which corresponds to a tuple variable
The "AS" may be dropped in most SQL implementations

Unspecified WHERE Clause

- Missing `WHERE` clause
  - Indicates no condition on tuple selection
- Effect is a `CROSS PRODUCT`
  - Result is all possible tuple combinations (Cartesian Product) result

Queries 9 and 10. Select all EMPLOYEE Ssns (Q9) and all combinations of EMPLOYEE Ssn and DEPARTMENT Dname (Q10) in the database.



```
Q9:    SELECT   Ssn
       FROM     EMPLOYEE;

Q10:   SELECT   Ssn, Dname
       FROM     EMPLOYEE, DEPARTMENT;
```

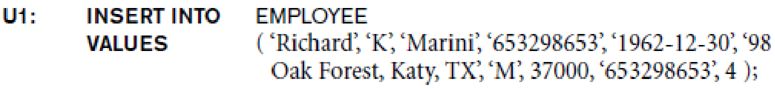
INSERT, DELETE, and UPDATE Statements in SQL

- Three commands used to modify the database:
  - `INSERT`, `DELETE`, and `UPDATE`
- `INSERT` typically inserts a tuple (row) in a relation (table)
- `UPDATE` may update a number of tuples (rows) in a relation (table) that satisfy the condition
- `DELETE` may also update a number of tuples (rows) in a relation (table) that satisfy the condition

53

INSERT

- In its simplest form, it is used to add one or more tuples to a relation
- Attribute values should be listed in the same order as the attributes were specified in the **CREATE TABLE** command
- Constraints on data types are observed automatically
- Any integrity constraints as a part of the DDL specification are enforced
- Specify the relation name and a list of values for the tuple. All values including nulls are supplied.

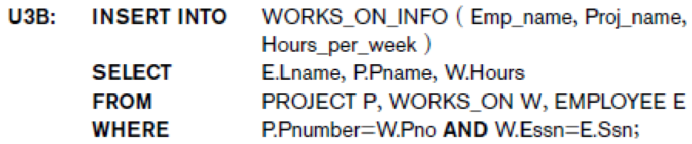


```
U1:    INSERT INTO    EMPLOYEE
       VALUES         ( 'Richard', 'K', 'Marini', '653298653', '1962-12-30', '98
                        Oak Forest, Katy, TX', 'M', 37000, '653298653', 4 );
```

54

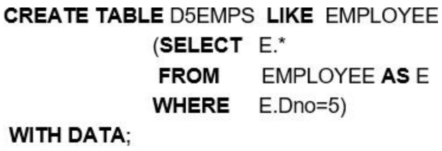
INSERT (Cont'd)

- The variation below inserts multiple tuples where a new table is loaded values from the result of a query.



```
U3B:    INSERT INTO     WORKS_ON_INFO ( Emp_name, Proj_name,
                        Hours_per_week )
        SELECT          E.Lname, P.Pname, W.Hours
        FROM            PROJECT P, WORKS_ON W, EMPLOYEE E
        WHERE           P.Pnumber=W.Pno AND W.Essn=E.Ssn;
```

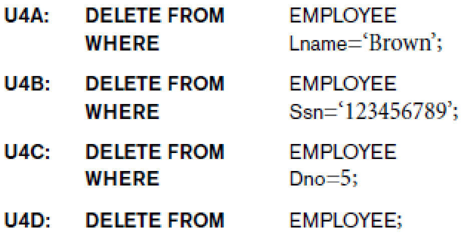
- Another variation of **INSERT** is used for bulk-loading of several tuples into tables
- A new table TNEW can be created with the same attributes as T and using LIKE and DATA in the syntax, it can be loaded with entire data.
- EXAMPLE:



```
CREATE TABLE D5EMPS  LIKE  EMPLOYEE
                    (SELECT   E.*
                     FROM     EMPLOYEE AS E
                    WHERE     E.Dno=5)
 WITH DATA;
```

DELETE

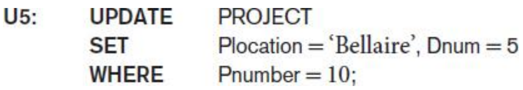
- Removes tuples from a relation
  - Includes a WHERE-clause to select the tuples to be deleted
  - Referential integrity should be enforced
  - Tuples are deleted from only **one table** at a time (unless CASCADE is specified on a referential integrity constraint)
  - A missing WHERE-clause specifies that **all tuples** in the relation are to be deleted; the table then becomes an empty table
  - The number of tuples deleted depends on the number of tuples in the relation that satisfy the WHERE-clause



```
U4A:    DELETE FROM     EMPLOYEE
        WHERE           Lname='Brown';

U4B:    DELETE FROM     EMPLOYEE
        WHERE           Ssn='123456789';

U4C:    DELETE FROM     EMPLOYEE
        WHERE           Dno=5;

U4D:    DELETE FROM     EMPLOYEE;
```

UPDATE

- Used to modify attribute values of one or more selected tuples
- A WHERE-clause selects the tuples to be modified
- An additional SET-clause specifies the attributes to be modified and their new values
- Each command modifies tuples **in the same relation**
- Referential integrity specified as part of DDL specification is enforced
- Example: Change the location and controlling department number of project number 10 to ‘Bellaire’ and 5, respectively



```
U5:   UPDATE   PROJECT
      SET      Plocation = 'Bellaire', Dnum = 5
      WHERE    Pnumber = 10;
```

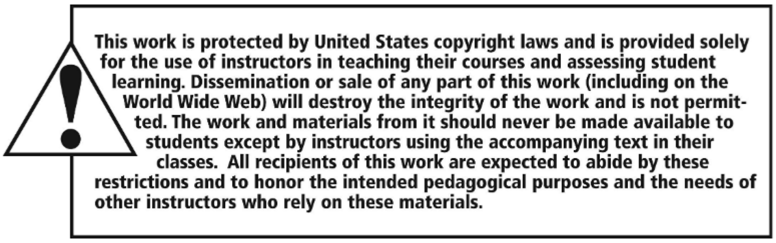
- Example: Give all employees in the ‘Research’ department a 10% raise in salary.

```
U6:UPDATE     EMPLOYEE
   SET        SALARY = SALARY *1.1
   WHERE      DNO  IN (SELECT     DNUMBER
                 FROM       DEPARTMENT
                 WHERE      DNAME='Research')
```

Pearson

57

Copyright



This work is protected by United States copyright laws and is provided solely for the use of instructors in teaching their courses and assessing student learning. Dissemination or sale of any part of this work (including on the World Wide Web) will destroy the integrity of the work and is not permitted. The work and materials from it should never be made available to students except by instructors using the accompanying text in their classes. All recipients of this work are expected to abide by these restrictions and to honor the intended pedagogical purposes and the needs of other instructors who rely on these materials.

Pearson

58