CS275 – Intro to Databases

The Relational Data Model - *Chap. 3*

How Is Data Retrieved and Manipulated?

Data Definition Language (DDL) statements used to define the database structure.

- CREATE - to create objects in the database
- ALTER - alters the structure of the database
- DROP - delete objects from the database

How Is Data Retrieved and Manipulated?

Data Manipulation Language (DML) statements used for managing data within schema objects.

SELECT - retrieve data from the a database

INSERT - insert data into a table

UPDATE - updates existing data within a table

DELETE - deletes all records from a table, the space for the records remain

An Example

- UNIVERSITY database
  - Information concerning students, courses, and grades in a university environment
- **Data records**
  - STUDENT
  - COURSE
  - SECTION
  - GRADE_REPORT
  - PREREQUISITE

An Example (cont'd.)

- Specify structure of records of each file by specifying **data type** for each **data element**
  - String of alphabetic characters
  - Integer
  - Etc.

Student(*name*: string, *snum*: integer, *class*: string, *major*: string)

Course(*cname*: string, *cnum*: string, *credits*: integer, *dept*: string)

Grade_Report(*snum*: integer, *sid*: integer, *grade*: string)

Section(*sid*: integer, *cnum*: string, *sem*: string, *year*: integer,
instr: string)

Etc.

An Example (cont'd.)

- Construct UNIVERSITY database
  - Store data to represent each student, course, section, grade report, and prerequisite as a record in appropriate file
- Relationships among the records
- Manipulation involves querying and updating

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

An Example (cont'd.)

- Examples of queries:
  - Retrieve the transcript
  - List the names of students who took the section of the ‘Database’ course offered in fall 2008 and their grades in that section
  - List the prerequisites of the ‘Database’ course

An Example (cont'd.)

- Examples of updates:
  - Change the class of ‘Smith’ to sophomore
  - Create a new section for the ‘Database’ course for this semester
  - Enter a grade of ‘A’ for ‘Smith’ in the ‘Database’ section of last semester

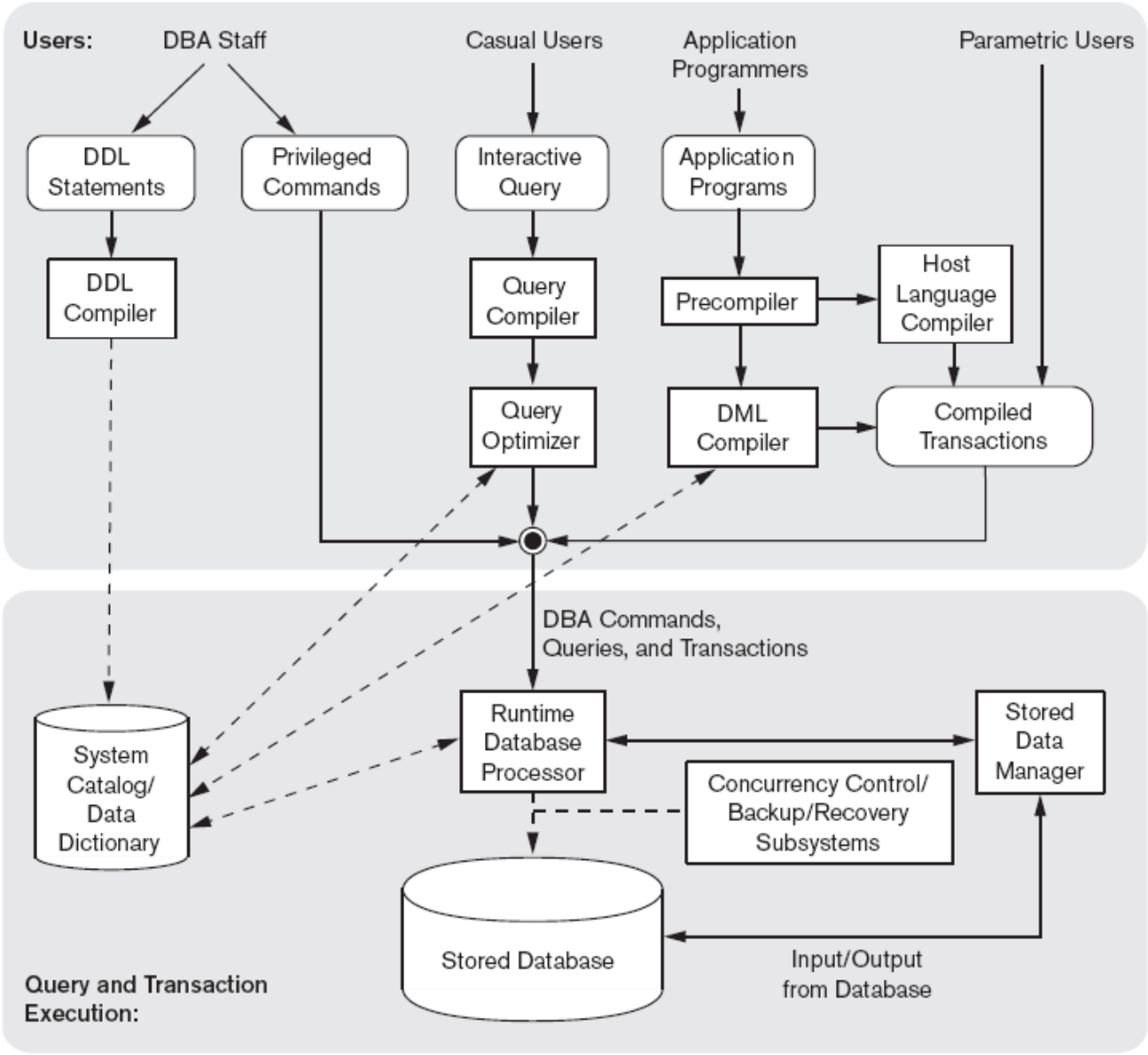


Figure 2.3
Component modules of a DBMS and their interactions.

Relational Model Concepts

- Collection of relations
- Table of values
  - Row
    - **Tuple** of related data values
    - How are the data values related?
  - Table name
    - What is the purpose of this?
  - Column names
    - **Attributes** providing meaning for the values

Domains, Attributes, Tuples, and Relations

- **Domain** D
  - Set of atomic values
  - Each value indivisible
- Specifying a domain
  - **Name**
  - **Data type**
  - **Format**
- Ex. Create domain ssn_type as char(9)

Domains, Attributes, Tuples, and Relations (cont'd.)

- **Relation schema** R
  - Denoted by $R(A_1, A_2, ..., A_n)$
  - Made up of a relation name R and a list of attributes, $A_1, A_2, ..., A_n$
- Attribute A_i
  - Name of a role played by some domain D in the relation schema R
- **Degree** (or **arity**) of a relation
  - Number of attributes n of its relation schema

Domains, Attributes, Tuples, and Relations (cont'd.)

- **Relation** (or **relation state**)
  - Set of ***n*-tuples** $r = \{t_1, t_2, ..., t_m\}$
  - Each n-tuple t
    - Ordered list of n values $t = <v_1, v_2, ..., v_n>$
    - Each value v_i, $1 \leq i \leq n$, is an element of dom(A_i) or is a special NULL value

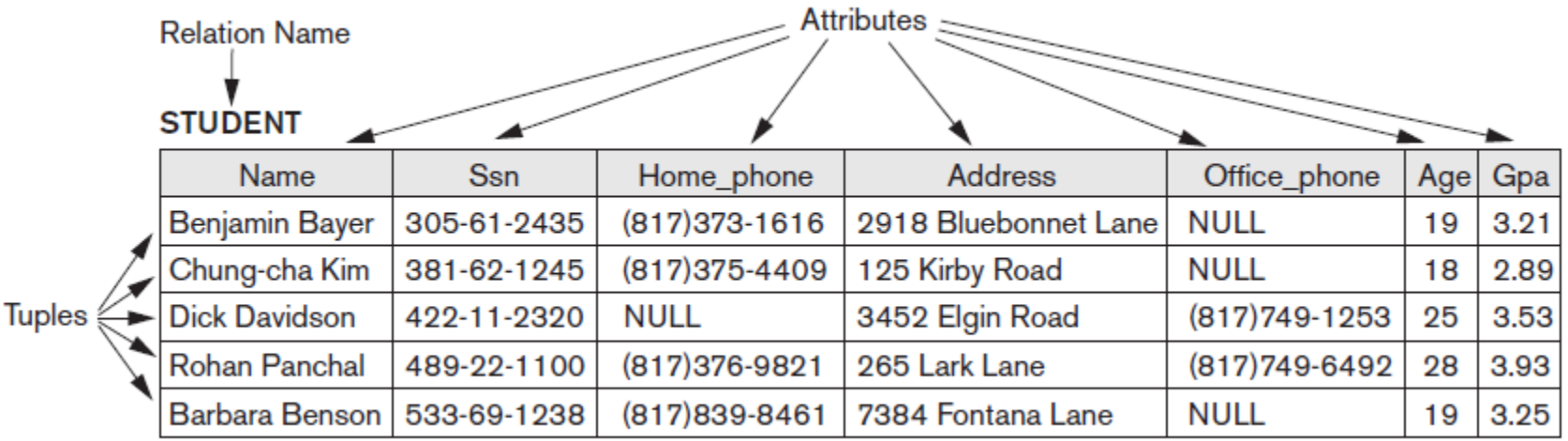
Domains, Attributes, Tuples, and Relations (cont'd.)

- Relation (or relation state) $r(R)$
  - **Subset** of the **Cartesian product** of the domains that define R:
    - $r(R) \subseteq (\text{dom}(A_1) \times \text{dom}(A_2) \times \ldots \times \text{dom}(A_n))$
- **Cardinality**
  - Total number of values in domain

Definition Summary

Informal Terms	***Formal Terms***
Table	Relation
Column	Attribute/Domain
Row	Tuple
Values in a Column	Domain
Table Definition	Schema of a Relation
Populated table	Extension

Relational Model Concepts (cont'd.)



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Characteristics of Relations

- Ordering of tuples in a relation
  - Relation defined as a set of tuples
  - Elements have no order among them
- Ordering of values within a tuple and an alternative definition of a relation
  - Order of attributes and values is not that important
  - As long as correspondence between attributes and values maintained

Characteristics of Relations (cont'd.)

- Alternative definition of a relation
  - Tuple considered as a set of (<attribute>, <value>) pairs
  - Each pair gives the value of the mapping from an attribute A_i to a value v_i from dom(A_i)
- Use the first definition of relation
  - Attributes and the values within tuples are ordered
  - Simpler notation

Characteristics of Relations (cont'd.)

- Values and NULLs in tuples
  - Each value in a tuple is atomic
  - **Flat relational model**
    - Composite and multivalued attributes not allowed
    - **First normal form** assumption
  - Multivalued attributes
    - Must be represented by separate relations
  - Composite attributes
    - Represented only by simple component attributes in basic relational model

1 NF

First Normal Form

- A relation is in 1NF if all attribute values are atomic: no repeating group, no composite attributes.
- Not in 1NF

DEPT_NO	MANAGER_NO	EMP_NO	EMP_NAME
D101	12345	20000 20001 20002	Carl Sagan Magic Johnson Larry Bird
D102	13456	30000 30001	Jimmy Carter Paul Simon

1 NF(Cont)

The corresponding relation in 1 NF:

DEPT_NO	MANAGER_NO	EMP_NO	EMP_NAME
D101	12345	20000	Carl Sagan
D101	12345	20001	Magic Johnson
D101	12345	20002	Larry Bird
D102	13456	30000	Jimmy Carter
D102	13456	30001	Paul Simon

Characteristics of Relations (cont'd.)

- NULL values
  - Represent the values of attributes that may be unknown or may not apply to a tuple
  - Meanings for NULL values
    - *Value unknown*
    - *Value exists but is not available (or withheld)*
    - *Attribute does not apply to this tuple (undefined)*
  - Be careful with NULL values

Relational Model Notation

- Relation schema R of degree n
  - Denoted by $R(A_1, A_2, ..., A_n)$
- Uppercase letters Q, R, S
  - Denote relation names
- Lowercase letters q, r, s
  - Denote relation states
- Letters t, u, v
  - Denote tuples

Relational Model Notation

- Name of a relation schema: STUDENT
  - Indicates the current set of tuples in that relation
- Notation: STUDENT(Name, Ssn, ...)
  - Refers only to relation schema
- Attribute *A* can be qualified with the relation name *R* to which it belongs
  - Using the dot notation *R.A*

Relational Model Constraints

- Constraints
  - Constraints are conditions that must hold on all valid relation instances.
  - Derived from the rules in the miniworld that the database represents.
- **Inherent model-based constraints** or **implicit constraints**
  - Inherent in the data model

Relational Model Constraints (cont'd.)

- **Schema-based constraints** or **explicit constraints**
  - Can be directly expressed in schemas of the data model
- **Application-based** or **semantic constraints** or **business rules**
  - Cannot be directly expressed in schemas
  - Expressed and enforced by application program

Domain Constraints

- Typically include:
  - Numeric data types for integers and real numbers
  - Characters
  - Booleans
  - Fixed-length strings
  - Variable-length strings
  - Date, time, timestamp
  - Money
  - Other special data types

Key Constraints and Constraints on NULL Values

- No two tuples can have the same combination of values for all their attributes.
- **Superkey**
  - No two distinct tuples in any state *r* of *R* can have the same value for SK
- **Key**
  - Superkey of R
  - Removing any attribute *A* from *K* leaves a set of attributes *K* that is not a superkey of *R* any more

Key Constraints and Constraints on NULL Values (cont'd.)

- Key satisfies two properties:
  - Two distinct tuples in any state of relation cannot have identical values for (all) attributes in key
  - Minimal superkey
    - Cannot remove any attributes and still have uniqueness constraint in above condition hold

Example: The CAR relation schema:

CAR(Engine#, PlateNo, Make, Model, Year)

has two keys Key1 = {Engine#}, Key2 = {PlateNo}, which are also superkeys. {PlateNo, Make} is a superkey but not a key.

Key Constraints and Constraints on NULL Values (cont'd.)

- **Candidate key**
  - Relation schema may have more than one key
- **Primary key** of the relation
  - Designated among candidate keys
  - Underline attribute
- Other candidate keys are designated as **unique keys**

Key Constraints and Constraints on NULL Values (cont'd.)

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

Figure 3.4
The CAR relation, with two candidate keys: License_number and Engine_serial_number.

Relational Databases and Relational Database Schemas

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

Relational Databases and Relational Database Schemas (cont'd.)

- **Invalid state**
  - Does not obey all the integrity constraints
- **Valid state**
  - Satisfies all the constraints in the defined set of integrity constraints IC

Integrity, Referential Integrity, and Foreign Keys

- **Entity integrity constraint**
  - No primary key value can be NULL
  - t[PK]≠ null for any tuple t in r(R)
- **Referential integrity constraint**
  - Specified between two relations
  - Maintains consistency among tuples in two relations
  - Used to specify a *relationship* among tuples in two relations: the **referencing relation** and the **referenced relation**.

Integrity, Referential Integrity, and Foreign Keys (cont'd.)

- **Foreign key** rules:
  - The attributes in FK have the same domain(s) as the primary key attributes PK
  - 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 the corresponding primary key PK in the referenced relation R2, or..

(2) a null.

In case (2), the FK in R1 should not be a part of its own primary key.

Integrity, Referential Integrity, and Foreign Keys (cont'd.)

- Diagrammatically display referential integrity constraints
  - Directed arc from each foreign key to the relation it references
- All integrity constraints should be specified on relational database schema

Update Operations, Transactions, and Dealing with Constraint Violations

- Operations of the relational model can be categorized into retrievals and updates
- Basic operations that change the states of relations in the database:
  - Insert
  - Delete
  - Update (or Modify)

Figure 3.6
One possible database state for the COMPANY relational database 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

Figure 3.6
One possible database state for the COMPANY relational database schema.

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

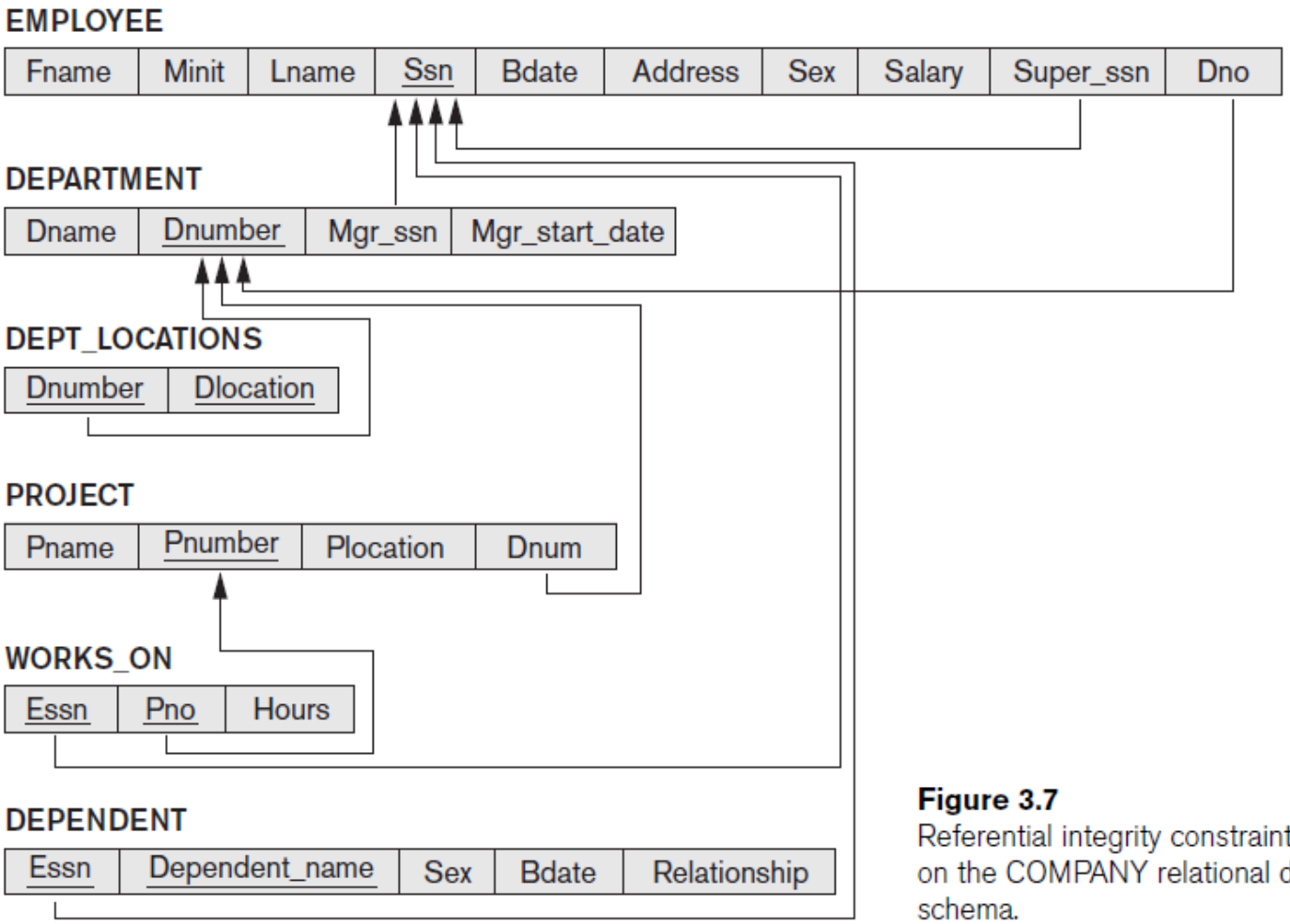


Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

The Delete/Update Operation

- Can violate referential integrity
  - If tuple being deleted/updated is referenced by foreign keys from other tuples
  - **Restrict**
    - Reject the deletion/update
  - **Cascade**
    - Propagate the modification to tuples that reference the tuple that is being modified
  - **Set null** or **set default**
    - Modify the referencing attribute values that cause the violation