Fundamentals of databases & Database design

IT 1400- Level 1 Semester 2
Lecture 6

Thilina Ranbaduge
tranbaduge@uom.lk

What we discuss Today ……..

- Relational Model Concepts
- Relational Model Constraints and Relational Database Schemas
- Update Operations and Dealing with Constraint Violations

Sets

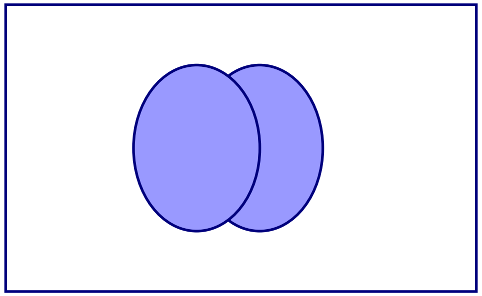
- A set is any well-defined list, collection, or class of objects.
- Elements or members of the set.
- $A = \{1, 3, 7, 10\}$
- $B= \{x \mid x \text{ is even}\}$

Subsets

- If every element in a set A is also a member of a set B, then A is called a subset of B.
- $A \subset B$ (Proper)

Union

- The union of sets A and B is the set of all elements which belong to A or to B or to both.
- $A \cup B$
- $S=\{a,b,c,d\}$ $T=\{f,b,d,g\}$ $S \cup T=?$



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 strength of the relational approach to data management comes from the formal foundation provided by the theory of relations.
- We review the essentials of the relational approach today.

Relational Model Concepts

- The model was first proposed by Dr. E.F. Codd of IBM 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 Ted Codd the coveted ACM Turing Award.

INFORMAL DEFINITIONS

- RELATION: A table of values
  - A relation may be thought of as a **set of rows**.
  - A relation may alternately be though of as a **set of columns**.
  - Each row represents a fact that corresponds to a real-world **entity** or **relationship**.
  - Each row has a value of an item or set of items that **uniquely identifies** that row in the table.
  - Sometimes row-ids or sequential numbers are assigned to identify the rows in the table.
  - Each column typically is called by its column name or column header or attribute name.

FORMAL DEFINITIONS

- A **Relation** may be defined in multiple ways.
- The **Schema** of a Relation: ***R*** (A1, A2, ...An)

Relation schema *R* is defined over **attributes** A1, A2, ...An

For Example -

CUSTOMER (Cust-id, Cust-name, Address, Phone#)

Here, CUSTOMER is a relation defined over the four attributes Cust-id, Cust-name, Address, Phone#, each of which has a **domain** or a set of valid values. For example, the domain of Cust-id is 6 digit numbers.

FORMAL DEFINITIONS

- A **tuple** is an ordered set of values
- Each **value** is derived from an appropriate **domain**.
- Each row in the CUSTOMER table may be referred to as a tuple in the table and would consist of four values.

- **<632895, "John Smith", "101 Main St. Atlanta, GA 30332", "(404) 894-2000">**

  is a tuple belonging to the CUSTOMER relation.

- A **relation** may be regarded as a ***set of tuples*** (rows).
- **Columns** in a table are also called **attributes** of the relation.

FORMAL DEFINITIONS

- A **domain** has a logical definition: e.g., "USA_phone_numbers" are the set of 10 digit phone numbers valid in the U.S.
- A **domain** may have a data-type or a format defined for it. The USA_phone_numbers may have a format: (ddd)-ddd-dddd where each d is a decimal digit. E.g., Dates have various formats such as monthname, date, year or yyyy-mm-dd, or dd mm,yyyy etc.
- An attribute designates the **role** played by the domain. E.g., the domain Date may be used to define attributes "Invoice-date" and "Payment-date".

FORMAL DEFINITIONS

- The relation is formed over the cartesian product of the sets; each set has values from a domain; that domain is used in a specific role which is conveyed by the attribute name.
- For example, attribute Cust-name is defined over the domain of strings of 25 characters. The role these strings play in the CUSTOMER relation is that of the name of customers.
- Formally,

  Given $R(A_1, A_2, ..., A_n)$

  $$\mathbf{r(R) \subset dom\ (A_1)\ X\ dom\ (A_2)\ X\ ....X\ dom(A_n)}$$
- R: schema of the relation
- r of R: a specific "value" or population of R.

- R is also called the **intension** of a relation
- r is also called the **extension** of a relation

FORMAL DEFINITIONS

- Let S1 = {0,1}
- Let S2 = {a,b,c}

- Let R $\subset$ S1 X S2

- Then for example: r(R) = {<0,a> , <0,b> , <1,c> }

  is one possible “**state**” or “population” or “**extension**” r of the relation R, defined over domains S1 and S2. It has three tuples.

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

Example

Relation name

Attributes

Tuples

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	Age	GPA
	Benjamin Bayer	305-61-2435	373-1616	2918 Bluebonnet Lane	null	19	3.21
	Katherine Ashly	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
	Charles Cooper	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

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.

- **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, ..., 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).

- **Values in a tuple**: All values are considered ***atomic*** (indivisible). A special **null value** is used to represent values that are **unknown** or **inapplicable** to certain tuples.

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

t = < (Name, Dick Davidson),(SSN, 422-11-2320),(HomePhone, null),(Address, 3452 Elgin Road),
(OfficePhone, 749-1253),(Age, 25),(GPA, 3.53)>

t = < (Address, 3452 Elgin Road),(Name, Dick Davidson),(SSN, 422-11-2320),(Age, 25),
(OfficePhone,749-1253),(GPA, 3.53),(HomePhone, null)>

CHARACTERISTICS OF RELATIONS

- <u>Notation:</u>

- We refer to **component values** of a tuple t by $t[A_i] = v_i$ (the value of attribute A_i for tuple t).

  Similarly, $t[A_u, A_v, ..., A_w]$ refers to the subtuple of t containing the values of attributes $A_u, A_v, ..., A_w$, respectively.

CHARACTERISTICS OF RELATIONS

STUDENT	Name	SSN	HomePhone	Address	OfficePhone	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
	Charles Cooper	489-22-1100	376-9821	265 Lark Lane	749-6492	28	3.93
	Katherine Ashly	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

t[Name] = Benjamin Bayer

Relational Integrity Constraints

- Constraints are *conditions* that must hold on *all* valid relation instances. There are three main types of constraints:
  1. **Key** constraints
  2. **Entity integrity** constraints
  3. **Referential integrity** constraints

Key Constraints

- Superkey of R: A set of attributes SK of R such that no two tuples *in any valid relation instance r(R)* will have the same value for SK. That is, for any distinct tuples t1 and t2 in r(R), t1[SK] ≠ t2[SK].

- Key of R: A "**minimal**" superkey; that is, a superkey K such that removal of any attribute from K results in a set of attributes that is not a superkey.

Example: The CAR relation schema:

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

has two keys Key1 = {State, Reg#}, Key2 = {SerialNo}, which are also superkeys.
{SerialNo, Make} is a superkey but *not* a key.

- If a relation has *several* **candidate keys**, one is chosen arbitrarily to be the **primary key**. The primary key attributes are *underlined*.

Key Constraints

CAR	LicenseNumber	EngineSerialNumber	Make	Model	Year
	Texas ABC-739	A69352	Ford	Mustang	96
	Florida TVP-347	B43696	Oldsmobile	Cutlass	99
	New York MPO-22	X83554	Oldsmobile	Delta	95
	California 432-TFY	C43742	Mercedes	190-D	93
	California RSK-629	Y82935	Toyota	Camry	98
	Texas RSK-629	U028365	Jaguar	XJS	98

Entity Integrity

- **Relational Database Schema**: A set S of relation schemas that belong to the same database. S is the *name* of the database.

  $$\mathbf{S = \{R_1, R_2, ..., R_n\}}$$

- **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.

  $$\mathbf{t[PK] \neq null \text{ for any tuple } t \text{ in } r(R)}$$

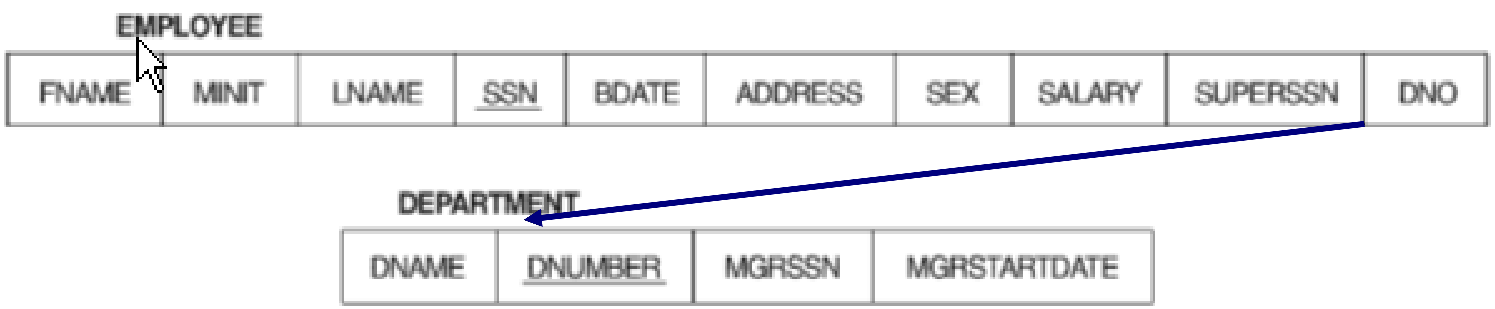
- Note: Other attributes of R may be similarly constrained to disallow null values, even though they are not members of the primary key.

Referential Integrity

- A constraint involving *two* relations (the previous constraints involve a *single* relation).
- Used to specify a *relationship* among tuples in two relations: the **referencing relation** and the **referenced relation**.
- 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 $R_1.FK$ to R_2.

DNO is a foreign key

t_1



t_2

t1[FK] = t2[PK].

Referential Integrity Constraint

<u>Statement of the constraint</u>

The value in the foreign key column (or columns) FK of the the **referencing relation** R_1 can be <u>either</u>:

(1) a value of an **existing primary key value** of the corresponding primary key PK in the **referenced relation** $R_{2,}$, or..

(2) a **null**.

In case (2), the FK in R_1 should <u>not</u> be a part of its own primary key.

Other Types of Constraints

Semantic Integrity Constraints:

- based on application semantics and cannot be expressed by the model per software engineer

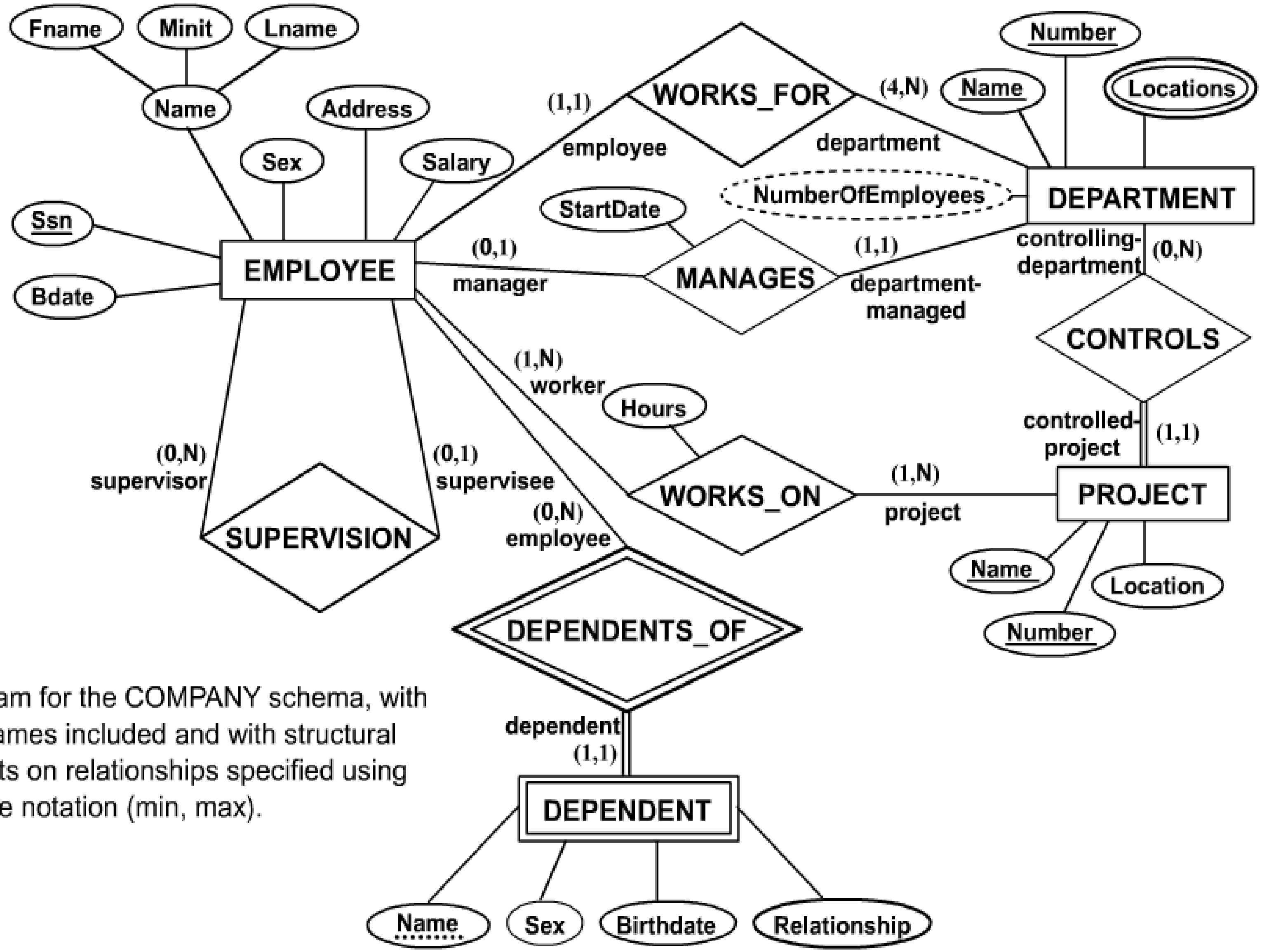
- E.g., “the max. no. of hours per employee for all projects he or she works on is 56 hrs per week”

- A *constraint specification language* may have to be used to express these

Converting ERD to Relational Schema

- Entity → Table
- Relation
  - 1:1 → put a foreign key in either side of entity
  - 1:N → put a foreign key in one side
  - N:N → create a table for it.
  - Ternary and over → in general make a table
- Attribute
  - Multi-valued attribute → Table
  - Make composite attribute flat

Alternative ER Notations



ER diagram for the COMPANY schema, with all role names included and with structural constraints on relationships specified using alternative notation (min, max).

EMPLOYEE

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO

DEPARTMENT

DNAME	DNUMBER	MGRSSN	MGRSTARTDATE

DEPT_LOCATIONS

DNUMBER	DLOCATION

PROJECT

PNAME	PNUMBER	PLOCATION	DNUM

WORKS_ON

ESSN	PNO	HOURS

DEPENDENT

ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John		Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin		Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia		Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer		Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh		Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce		English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad		Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James		Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

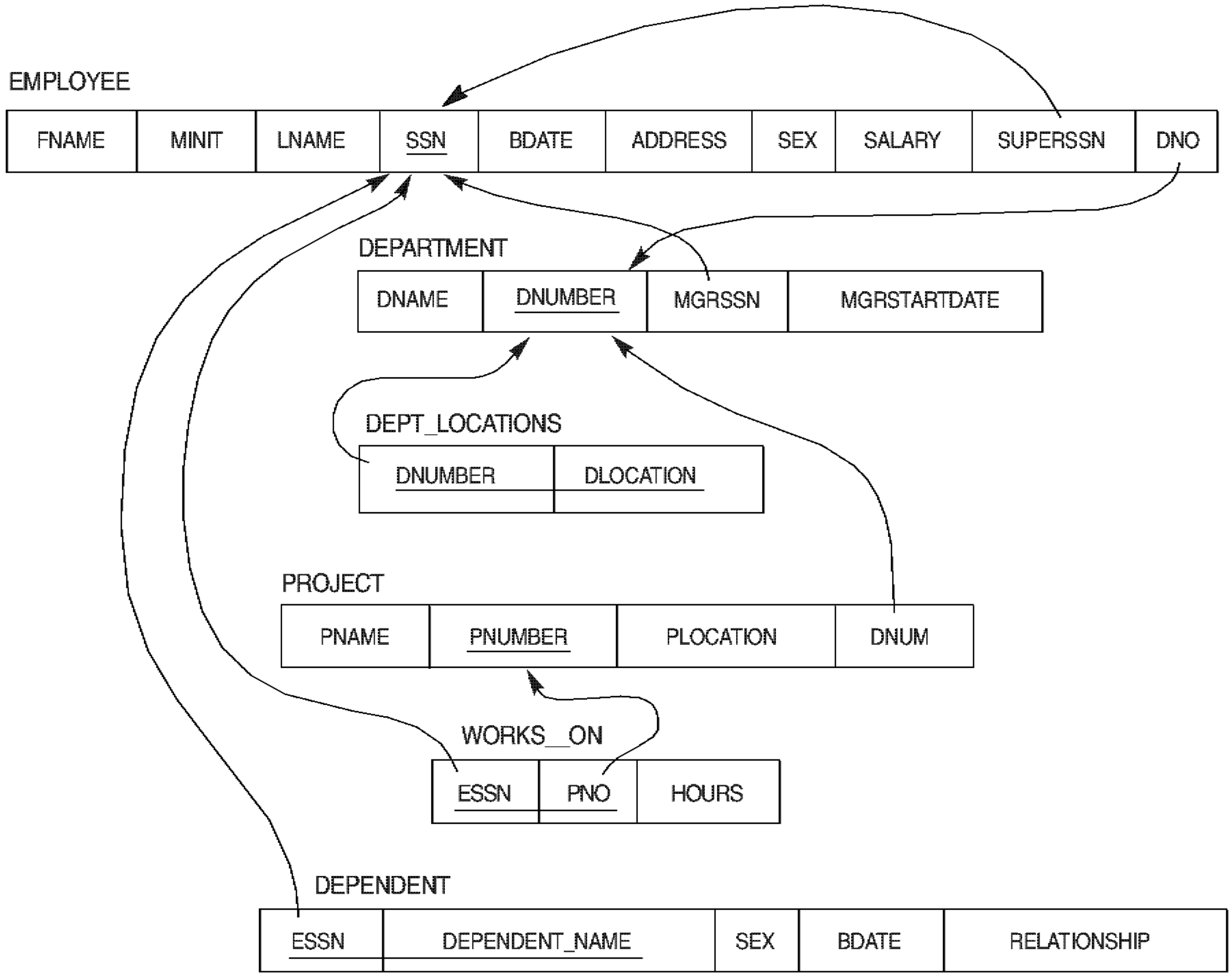
DEPT_LOCATIONS	DNUMBER	DLOCATION
		Houston
		Stafford
		Bellaire
		Sugarland

DEPARTMENT	DNAME	DNUMBER	MGRSSN	MGRSTARTDATE
	Research	5	333445555	1988-05-22
	Administration	4	987654321	1995-01-01
	Headquarters	1	888665555	1981-06-19

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


EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS__ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP

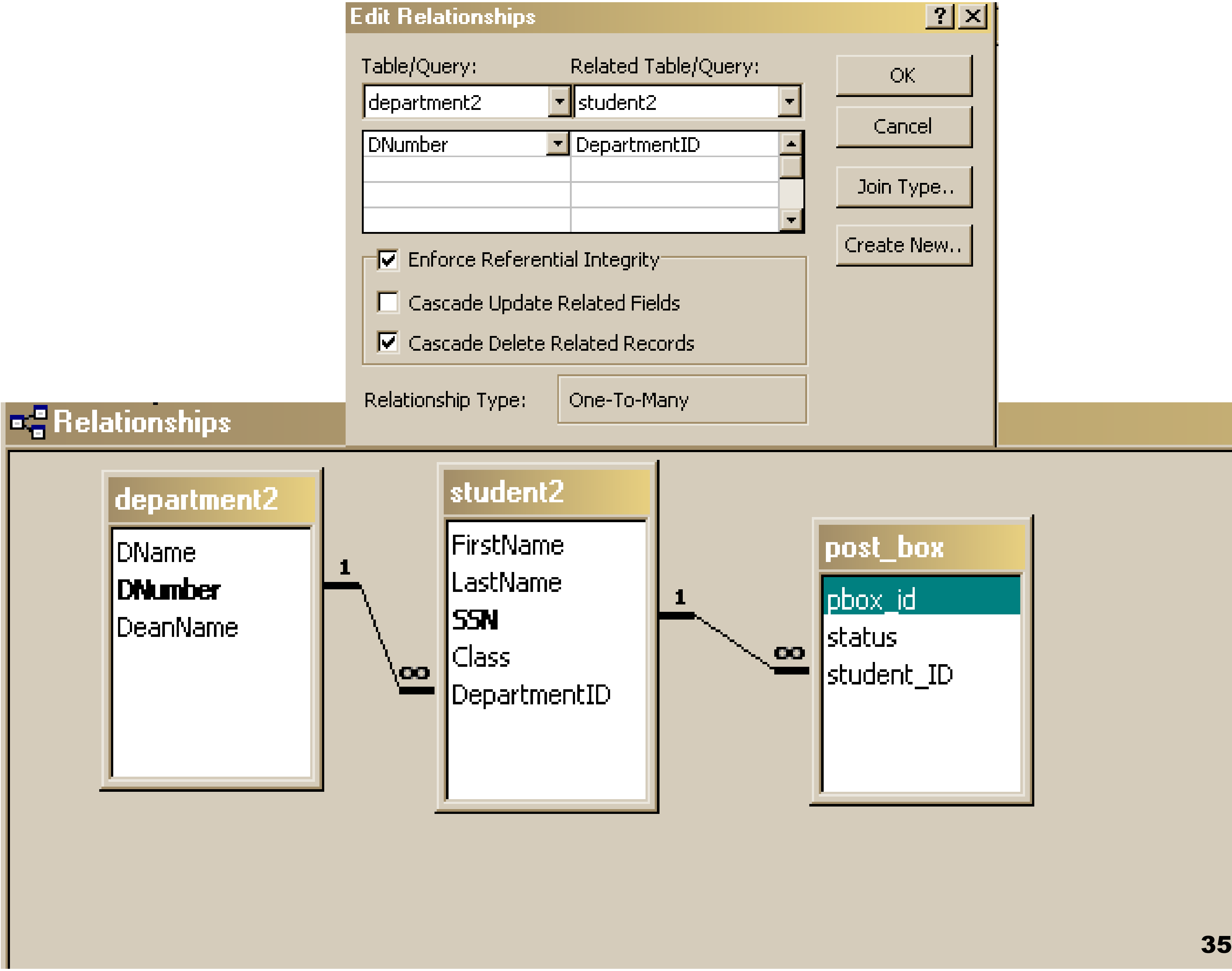
Update Operations on Relations

- 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.

Update Operations on Relations

- In case of integrity violation, several actions can be taken:
  - Cancel the operation that causes the violation (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


Edit Relationships
Table/Query:
Related Table/Query:
department2
student2
DNumber
DepartmentID
OK
Cancel
Join Type..
Create New..
Enforce Referential Integrity
Cascade Update Related Fields
Cascade Delete Related Records
Relationship Type:
One-To-Many
Relationships
department2
DName
DNumber
DeanName
student2
FirstName
LastName
SSN
Class
DepartmentID
post_box
pbox_id
status
student_ID
1
1

In-Class Exercise

Consider the following relations for a database that keeps track of student enrollment in courses and the books adopted for each course:

STUDENT(SSN, Name, Major, Bdate)

COURSE(Course#, Cname, Dept)

Quarter(Quarter)

ENROLL(SSN, Course#, Quarter, Grade)

BOOK_ADOPTION(Course#, Quarter, Book_ISBN)

TEXT(Book_ISBN, Book_Title, Publisher, Author)

Draw a relational schema diagram specifying the foreign keys for this schema.

References

- *Chapter 6 : Fundamentals of Database Systems*

(4th Edition) By Remez Elmasri & Shamkant B. Navathe

Questions ???