Modeling Recursive Relations

- **Modeling Recursive Relations**

Modeling Recursive Relations

How can we expand our E-R Model of EMPLOYEE and DEPARTMENT to represent the relationship between EMPLOYEE and his manager?

"Each EMPLOYEE may be managed by one and only one MANAGER."
"Each MANAGER may be the manager of one or more EMPLOYEEs."

Modeling Recursive Relations

Some sample instances of this relationship might be:

EMPLOYEE	MANAGER
John Brown	Mary Smith
Bob Phillips	Mary Smith
Mary Smith	Jim Jones
Jim Jones	

Every manager is also an employee. So manager is not a new entity, but just a subset of the instances of the entity EMPLOYEE.

Modeling Recursive Relations

Let us rewrite the relationshipsentences;

"Each EMPLOYEE may be managed by one and only one EMPLOYEE."
"Each EMPLOYEE may be the manager of one or more EMPLOYEEs."

This is a many-to-one relationship which is optional in both directions.

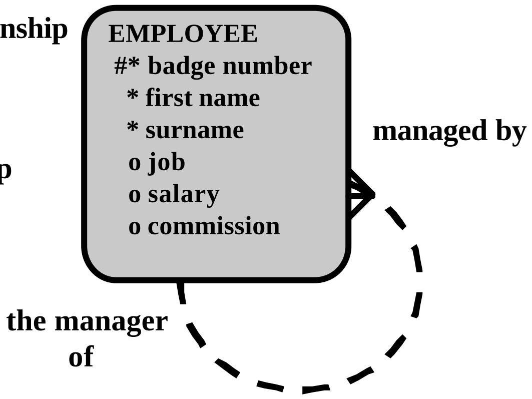
It is same as the relationship between two distinct entities.

Modeling Recursive Relations

Diagram a Recursive Relationship

We can diagram the relationship with the same techniques.

A recursive relationship is always modelled with a loop or a bear's ear.



Modeling Recursive Relations

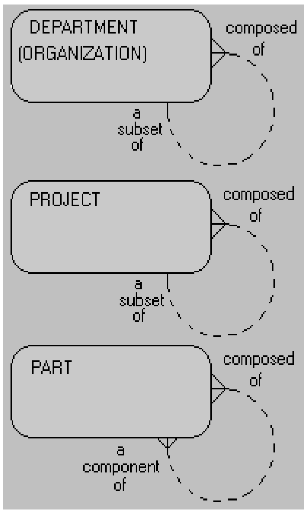
Which of the following entities do you think might be recursive?

A. PART

B. PROJECT

C. DEPARTMENT

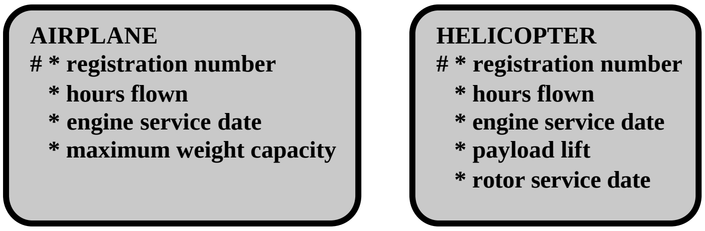
D. All of the above



● Identify Subtypes and Supertypes

Identify Subtypes and Supertypes

Often we will encounter entities which are <u>mutually exclusive</u>, but have common attributes. For example, an AIRPLANE entity and a HELICOPTER entity are mutually exclusive-- an airplane cannot also be helicopter, and a helicopter cannot be an airplane.

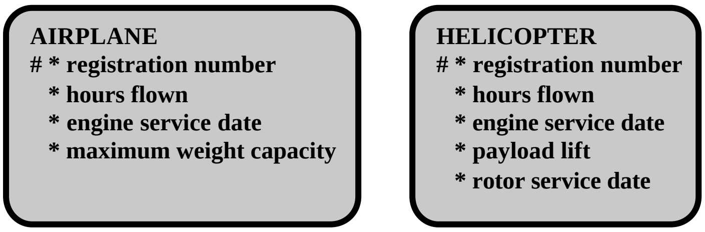


AIRPLANE
# * registration number
*** hours flown**
*** engine service date**
*** maximum weight capacity**

HELICOPTER
# * registration number
*** hours flown**
*** engine service date**
*** payload lift**
*** rotor service date**

Identify Subtypes and Supertypes

However, an AIRPLANE entity and HELICOPTER entity do share attributes in common- such as registration number, hours flown, and engine service date.



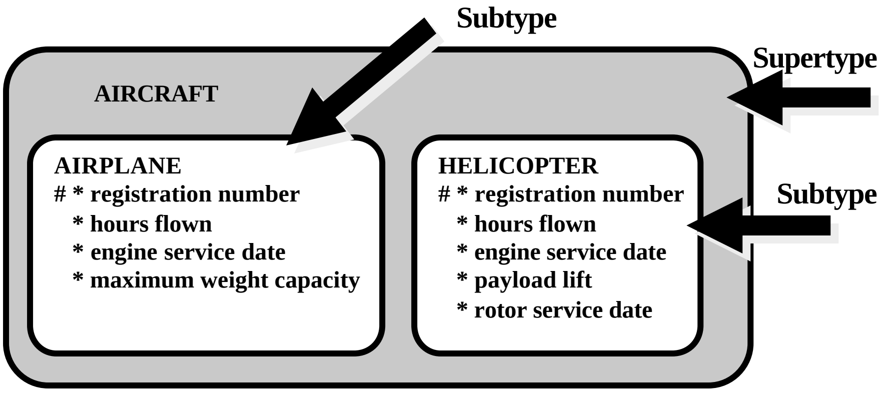
AIRPLANE
# * registration number
* hours flown
* engine service date
* maximum weight capacity

HELICOPTER
# * registration number
* hours flown
* engine service date
* payload lift
* rotor service date

Identify Subtypes and Supertypes

Diagram Subtypes

Exclusive entities can be modelled using a supertype and two or more subtypes. For example, we could define a supertype AIRCRAFT with subtypes of AIRPLANE and HELICOPTER.

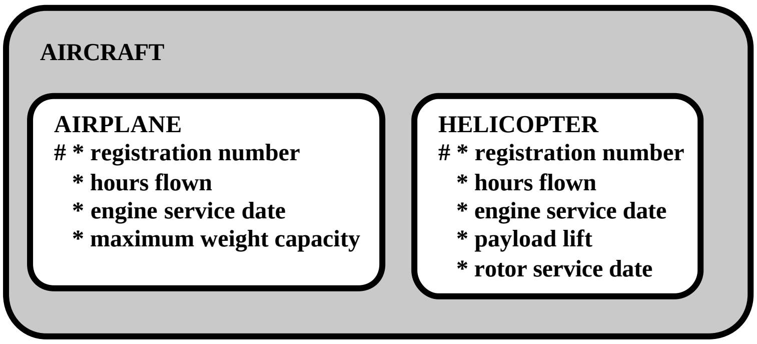


AIRPLANE
# * registration number
* hours flown
* engine service date
* maximum weight capacity

HELICOPTER
# * registration number
* hours flown
* engine service date
* payload lift
* rotor service date

Identify Subtypes and Supertypes

All instances of a supertype must belong to one and only one of the subtype entities. Can an AIRCRAFT be both an AIRPLANE and a HELICOPTER?



AIRCRAFT

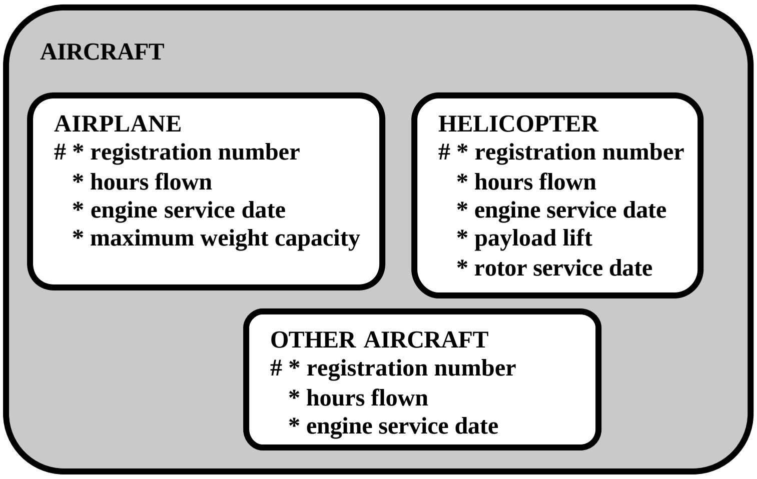
AIRPLANE
# * registration number
  * hours flown
  * engine service date
  * maximum weight capacity

HELICOPTER
# * registration number
  * hours flown
  * engine service date
  * payload lift
  * rotor service date

What about gliders, ultralights, and military ones? These are instances of AIRCRAFT which are not AIRPLANEs or HELICOPTERs.

Identify Subtypes and Supertypes

To accommodate these other AIRCRAFT, add a third subtype called OTHER AIRCRAFT



AIRCRAFT

AIRPLANE
# * registration number
  * hours flown
  * engine service date
  * maximum weight capacity

HELICOPTER
# * registration number
  * hours flown
  * engine service date
  * payload lift
  * rotor service date

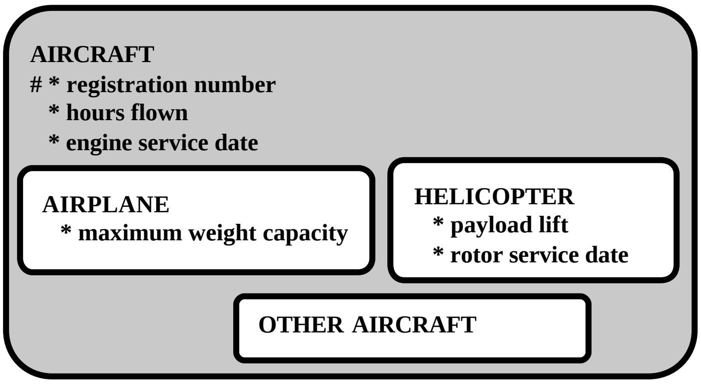
OTHER AIRCRAFT
# * registration number
  * hours flown
  * engine service date

Now the subtypes of AIRCRAFT form a complete, non-overlapping set.

Identify Subtypes and Supertypes

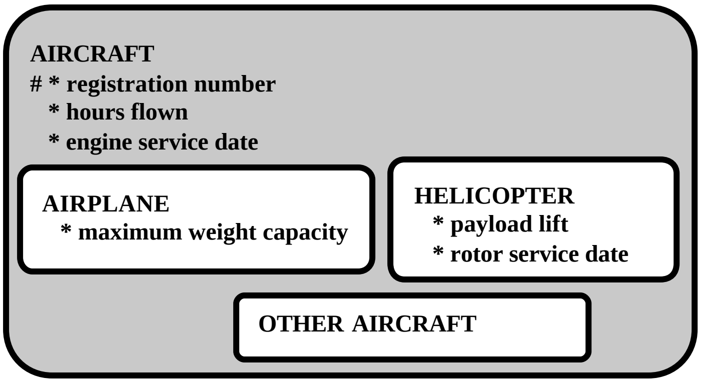
Attributes of Subtypes

A supertype can have attributes of its own. Each subtype will inherit the attributes of its supertype . Attributes shared by the subtypes should be defined at the supertype level. We can reassign the attributes shared by the subtypes HELICOPTER, AIRPLANE, and OTHER AIRCRAFT, to their supertype AIRCRAFT.



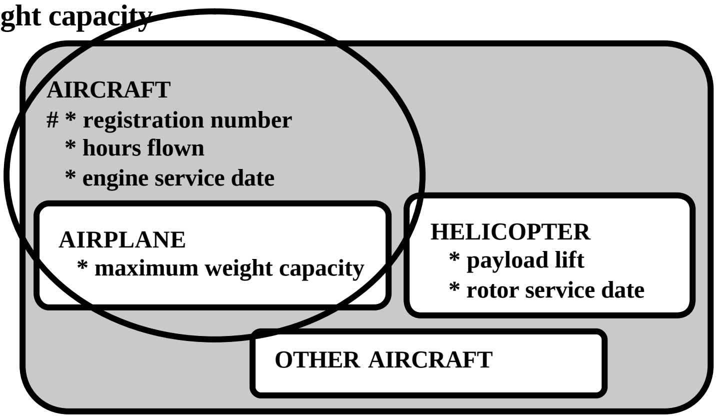
Identify Subtypes and Supertypes

For each instance of a subtype, the attributes of the supertype and the attributes specific to that subtype must be defined.



Identify Subtypes and Supertypes

For example, each instance of an AIRPLANE has the following attributes:
#* registration number
*** hours flown**
*** service date**
*** maximum weight capacity**



AIRCRAFT
# * registration number
*** hours flown**
*** engine service date**

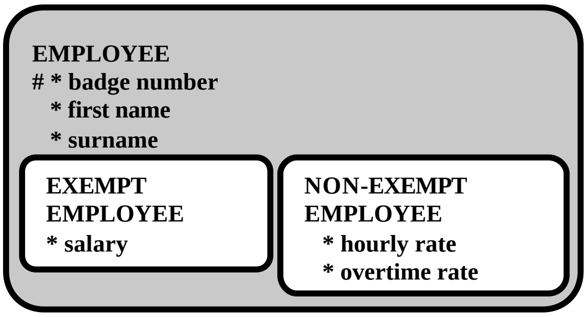
AIRPLANE
*** maximum weight capacity**

HELICOPTER
*** payload lift**
*** rotor service date**

OTHER AIRCRAFT

Identify Subtypes and Supertypes

What attributes exist for each instance of an EXEMPT EMPLOYEE?



EMPLOYEE
# * badge number
*** first name**
*** surname**

EXEMPT EMPLOYEE
*** salary**

NON-EXEMPT EMPLOYEE
*** hourly rate**
*** overtime rate**

A. * salary

B. # * badge number
*** salary**

C. # * badge number
*** first name**
*** surname**
*** salary**

D. # * badge number
*** first name**
*** surname**
*** hourly rate**
*** overtime rate**

Identify Subtypes and Supertypes

Relationships of Subtypes

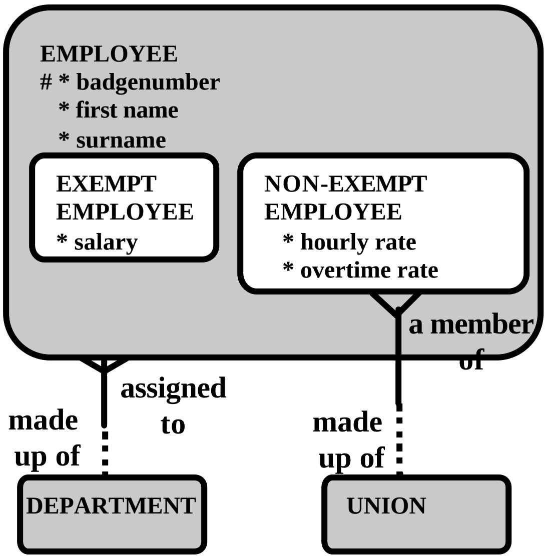
Both the supertype and its subtypes may have relationships to other entities.

Identify Subtypes and Supertypes

The EMPLOYEE supertype has a relationship with the entity DEPARTMENT.

"Each EMPLOYEE must be assigned to one and only one DEPARTMENT."

The relationship to DEPARTMENT applies to all instances of the supertype: all EXEMPT EMPLOYEEs and all NON-EXEMPT EMPLOYEEs. Subtypes inherit the relationship of their supertypes.



EMPLOYEE
# * badgenumber
* first name
* surname

EXEMPT EMPLOYEE
* salary

NON-EXEMPT EMPLOYEE
* hourly rate
* overtime rate

assigned to

made up of

DEPARTMENT

a member of

made up of

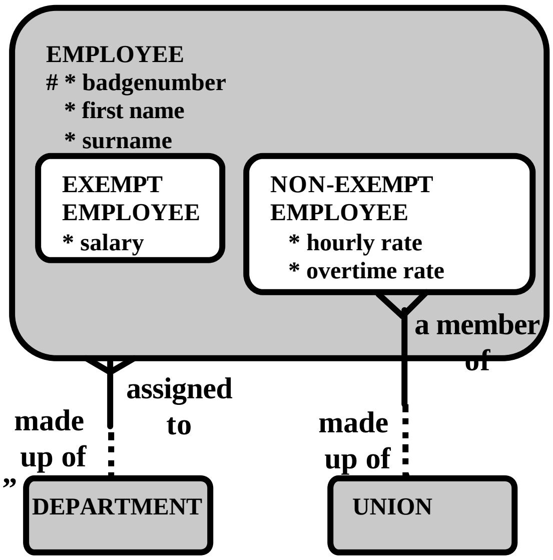
UNION

Identify Subtypes and Supertypes

A subtype may alsohave relationships of its own. Only non-exempt employees belong to the unions. So only the NON-EXEMPT EMPLOYEE subtype has a relationship with the UNION entity:

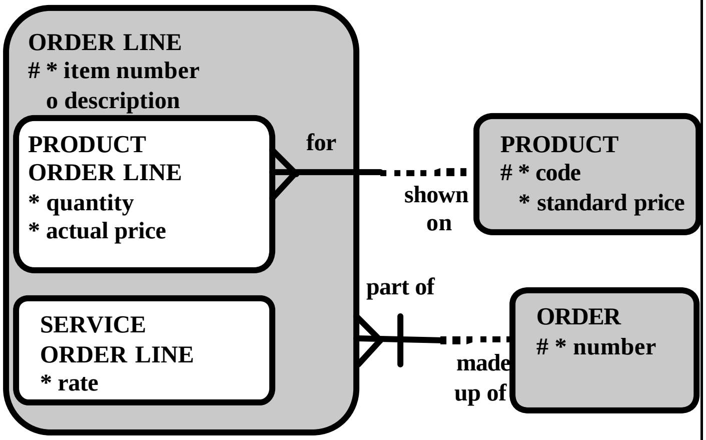
“Each NON-EXEMPT EMPLOYEE must be a member of one and only one UNION.”

“Each UNION may be made up of one or more NON-EXEMPT EMPLOYEEs”



Identify Subtypes and Supertypes

What entities does PRODUCT ORDER LINE have relationships with?

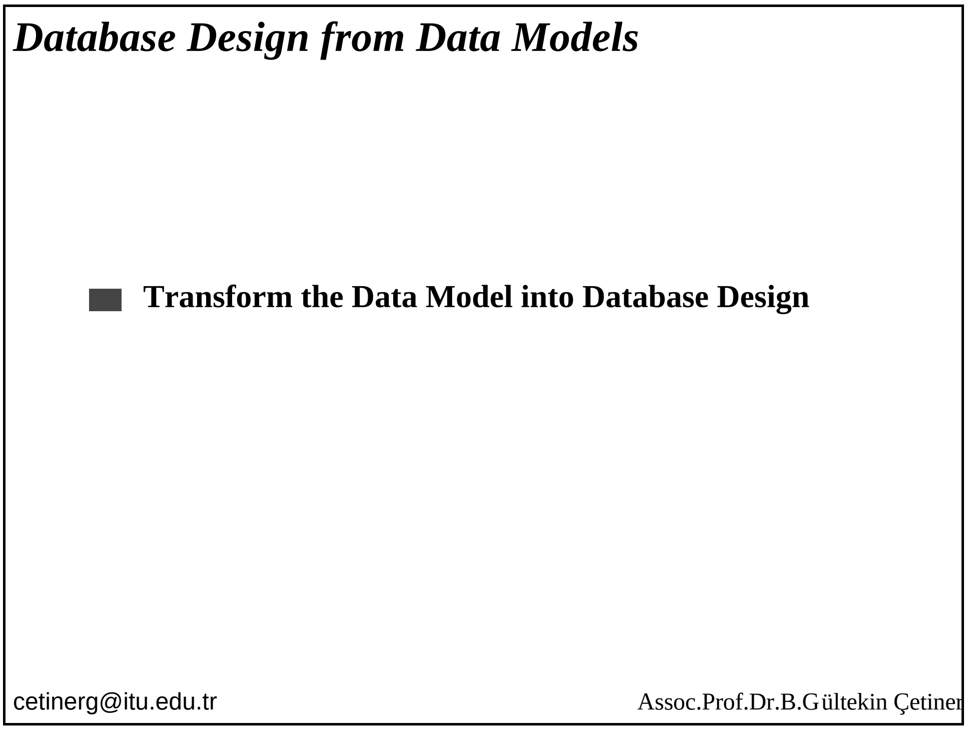


A. ORDER

B. PRODUCT ORDER

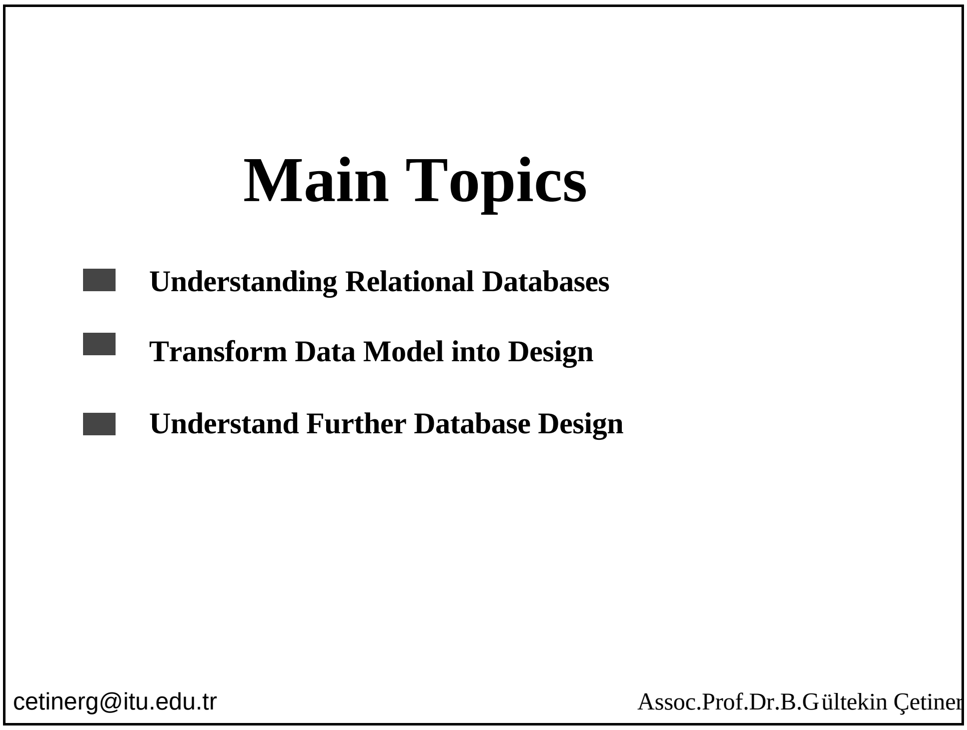
C. ORDER LINE SERVICE ORDER LINE

D. ORDER LINE PRODUCT



Database Design from Data Models

- **Transform the Data Model into Database Design**



Main Topics

- **Understanding Relational Databases**
- **Transform Data Model into Design**
- **Understand Further Database Design**

- **Database Design**
- **Relational Tables**
- **Primary Keys**
- **Foreign Keys**
- **SQL-Structured Query Language**

Understanding Relational Databases

Understanding Relational Databases

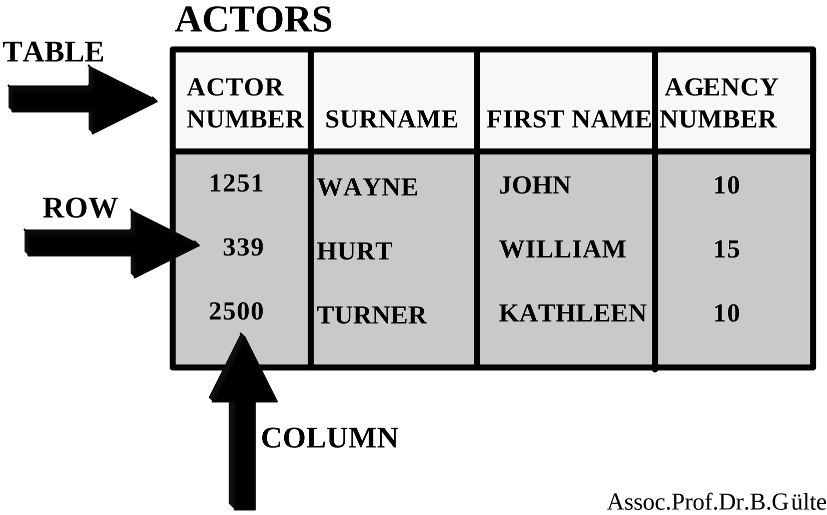
Topic Objectives

This topic introduces Database Design and Relational Database concepts. At the end of this topic, wou will be able to;

- **explain how Database Design fits into the Database Development Process**
- **identify the structures used in relational databases: tables and columns, primary and foreign keys**

Understanding Relational Databases

Here is an example of a relational table called ACTOR:



ACTORS

ACTOR NUMBER	SURNAME	FIRST NAME	AGENCY NUMBER
1251	WAYNE	JOHN	10
339	HURT	WILLIAM	15
2500	TURNER	KATHLEEN	10

Understanding Relational Databases

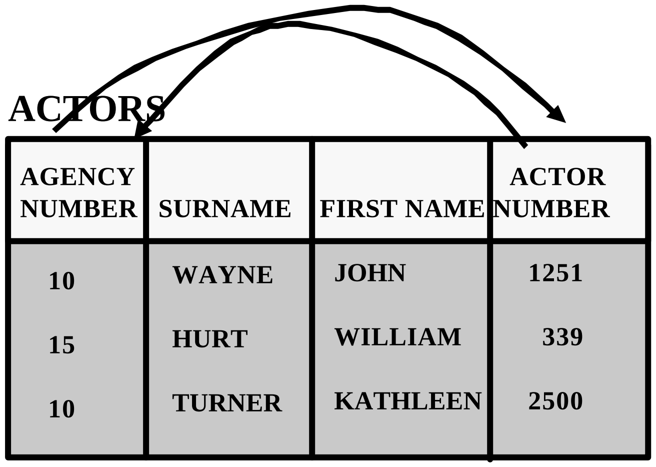
One of the main principles of relational DBMSs is that all programs which retrieve data from the table must use column names: no applications can reference “the third column.”

Understanding Relational Databases

In relational DBMSs, the physical order of the rows does not give them any meaning. A row means the same thing whether it is first, last, or anywhere else in the table, and whether it precedes or follows some other row.

Understanding Relational Databases

Suppose that the columns "ACTOR_NUMBER" and "AGENCY NUMBER" have been changed by redefining table structure, <u>the programs and queries will not be affected at all.</u>



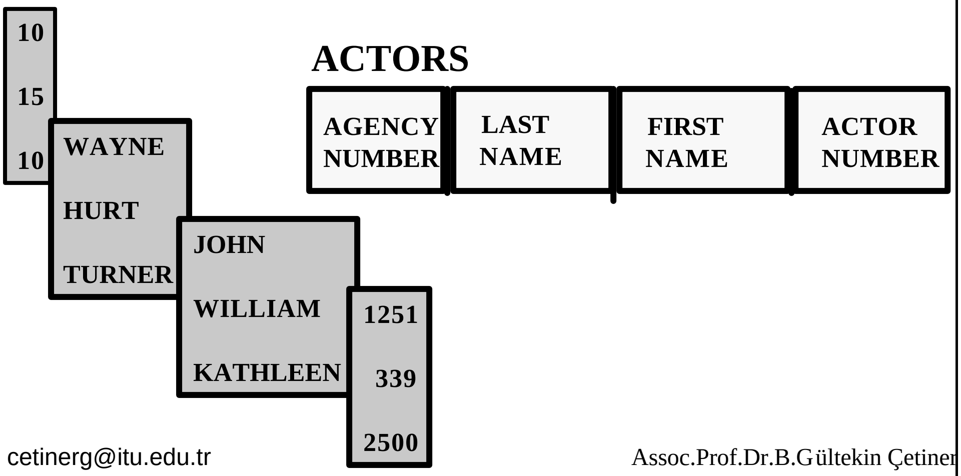
ACTORS

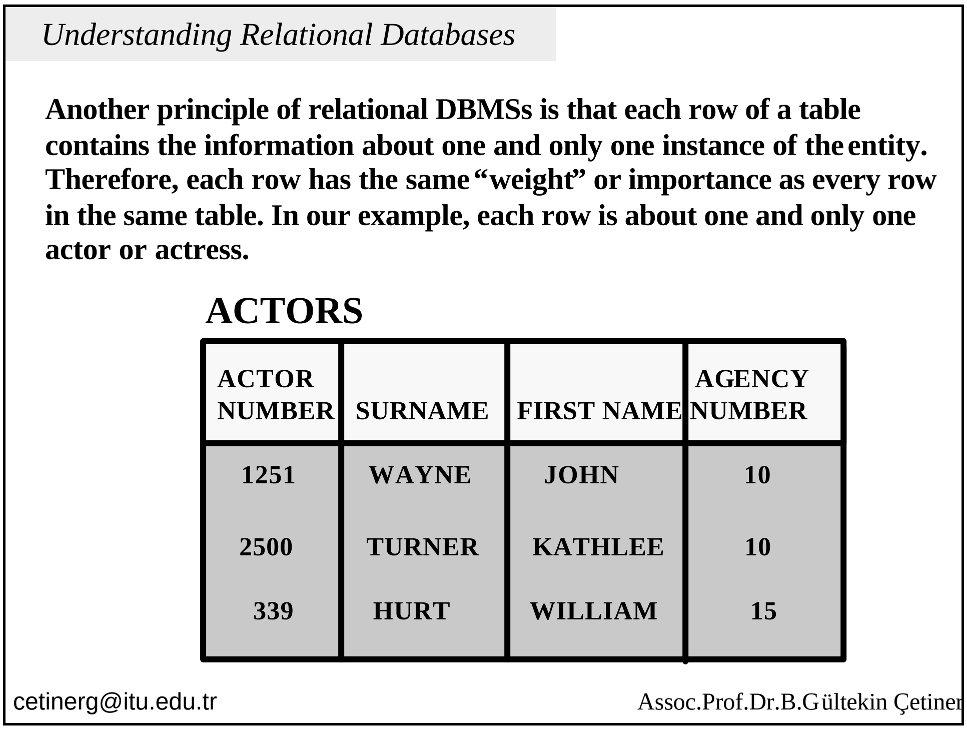
AGENCY NUMBER	SURNAME	FIRST NAME	ACTOR NUMBER
10	WAYNE	JOHN	1251
15	HURT	WILLIAM	339
10	TURNER	KATHLEEN	2500

Understanding Relational Databases

No meaning should be lost when the order of columns is changed.

(Select ACTOR_NUMBER, FIRST_NAME FROM ACTORS)

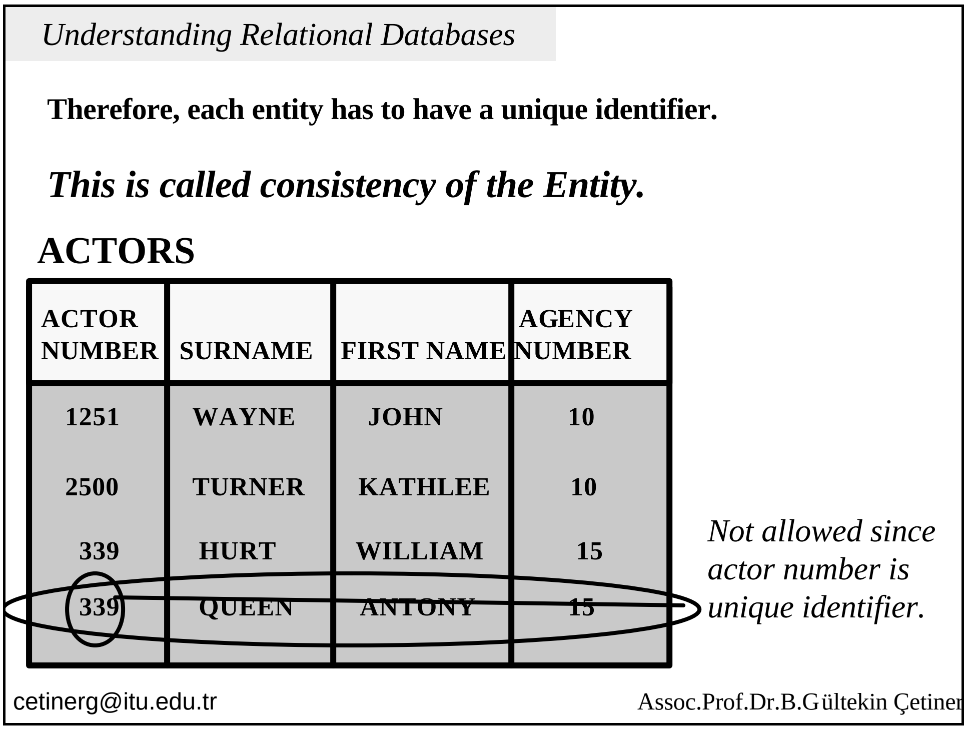




Another principle of relational DBMSs is that each row of a table contains the information about one and only one instance of the entity. Therefore, each row has the same "weight" or importance as every row in the same table. In our example, each row is about one and only one actor or actress.

ACTORS

ACTOR NUMBER	SURNAME	FIRST NAME	AGENCY NUMBER
1251	WAYNE	JOHN	10
2500	TURNER	KATHLEE	10
339	HURT	WILLIAM	15



Therefore, each entity has to have a unique identifier.

This is called consistency of the Entity.

ACTORS

ACTOR NUMBER	SURNAME	FIRST NAME	AGENCY NUMBER
1251	WAYNE	JOHN	10
2500	TURNER	KATHLEE	10
339	HURT	WILLIAM	15
339	QUEEN	ANTONY	15

Not allowed since actor number is unique identifier.

Understanding Relational Databases

Another rule:
No meaning should be lost when the order of rows is changed.

(Select ACTOR_NUMBER, FIRST_NAME FROM ACTORS where ACTOR_NUMBER=1251)

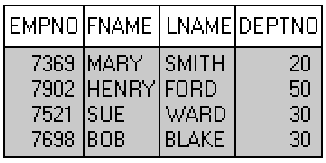
ACTORS

ACTOR NUMBER	SURNAME	FIRST NAME	AGENCY NUMBER
2500	TURNER	KATHLEE	10
1251	WAYNE	JOHN	10
339	HURT	WILLIAM	15

Understanding Relational Databases

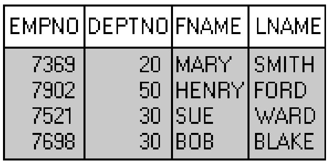
In a relational table, the physical order of the rows and columns is immaterial. Which of the following tables are equivalent?

Table 1



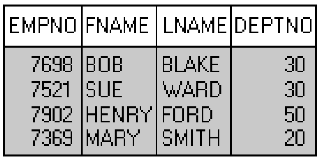
EMPNO	FNAME	LNAME	DEPTNO
7369	MARY	SMITH	20
7902	HENRY	FORD	50
7521	SUE	WARD	30
7698	BOB	BLAKE	30

Table 2



EMPNO	DEPTNO	FNAME	LNAME
7369	20	MARY	SMITH
7902	50	HENRY	FORD
7521	30	SUE	WARD
7698	30	BOB	BLAKE

Table 3



EMPNO	FNAME	LNAME	DEPTNO
7698	BOB	BLAKE	30
7521	SUE	WARD	30
7902	HENRY	FORD	50
7369	MARY	SMITH	20

A. 1

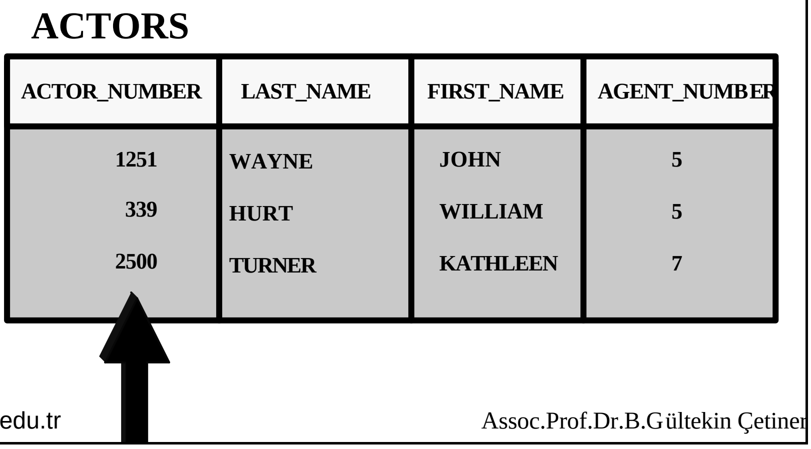
B. 1,2

C. 1,3

D. 1,2,3

Primary Keys

A Primary Key (PK) is a column or set of columns that uniquely identifies each row in a table. Each table must have a primary key. The primary key of the ACTORS table is ACTOR_NUMBER.

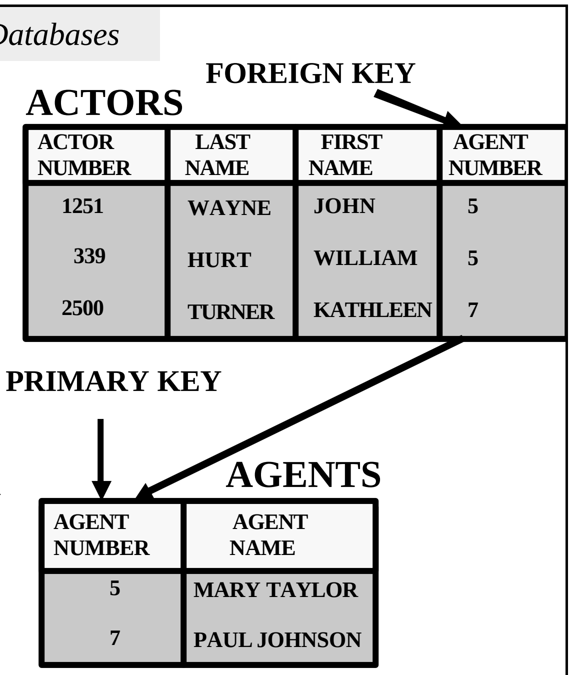


ACTORS

ACTOR_NUMBER	LAST_NAME	FIRST_NAME	AGENT_NUMBER
1251	WAYNE	JOHN	5
339	HURT	WILLIAM	5
2500	TURNER	KATHLEEN	7

FOREIGN KEYS

A Foreign Key (FK) is a column or combination of columns in one table that refers to a primary key in the same or another table. AGENT_NUMBER is a foreign key in the ACTORS Table and refers to AGENT_NUMBER column in the AGENTS Table.



FOREIGN KEY

ACTORS

ACTOR NUMBER	LAST NAME	FIRST NAME	AGENT NUMBER
1251	WAYNE	JOHN	5
339	HURT	WILLIAM	5
2500	TURNER	KATHLEEN	7

PRIMARY KEY

AGENTS

AGENT NUMBER	AGENT NAME
5	MARY TAYLOR
7	PAUL JOHNSON

Structured Query Language -SQL

Structured Query Language (SQL) is the internationally accepted standard language for querying and manipulating data in relational databases. The following SQL statement retrieves the values of ACTOR_NUMBER, LAST_NAME, FIRST_NAME, and AGENT_NUMBER from the ACTORS Table for Actor Number 350.

```
SELECT ACTOR_NUMBER,
LAST_NAME,FIRST_NAME,AGENT_NUMBER
  FROM ACTORS
    WHERE ACTOR_NUMBER=350
```

A primary key must uniquely identify each row of a table. What column or combination of columns could serve as the primary key of the CATALOG_ITEMS Table?

CATALOG_ITEMS

PRODUCT NUMBER	VENDOR NUMBER	PACKAGE QUANTITY	ITEM PRICE
99	5	5	$25,50
99	6	10	$15,35
99	5	20	$23,00
102	5	5	$25,00
103	5	3	$5,00

A. **PRODUCT_NUMBER**

B. **PRODUCT_NUMBER, ITEM_PRICE**

C. **ITEM_PRICE**

D. **PRODUCT_NUMBER, VENDOR_NUMBER, PACKAGE_QUANTITY**

Mapping (Transforming) Data Model into Database Design

Mapping Entities

Mapping Attributes

Mapping Unique Attributes

Mapping Relationships

Mapping Data Scales

Mapping (Transforming) Data Model into Database Design

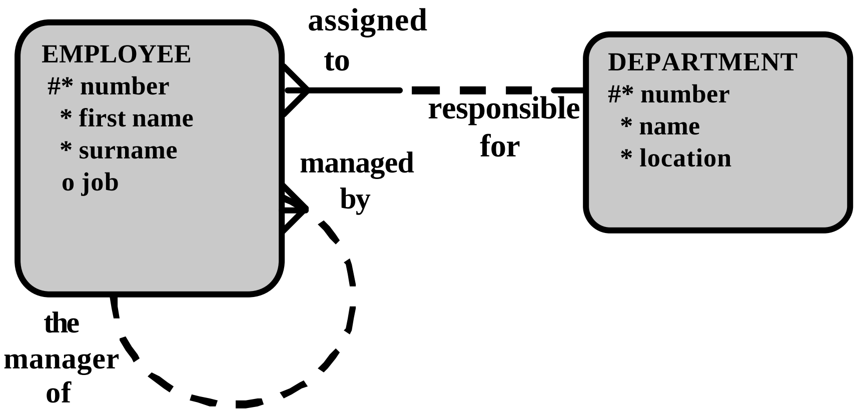
Mapping (Transforming) Data Model into Database Design

Mapping an E-R Diagram to a relational database physical design is fairly straightforward. Each part of the E-R Diagram translates directly into a part of a relational database design, as follows :

E-R MODEL	RELATIONAL DATABASE DESIGN
Entity →	**Table**
Attribute →	**Column**
Unique Identifier →	**Primary Key**
Relationship →	**Foreign Key**

Mapping (Transforming) Data Model into Database Design

Let’s map the following E-R Diagram to a relational database design.



Mapping (Transforming) Data Model into Database Design

Map Entities

First we will map the entities to tables.

E-R MODEL	RELATIONAL DATABASE DESIGN
Entity	**Table**
Attribute	**Column**
Unique Identifier	**Primary Key**
Relationship	**Foreign Key**

Mapping (Transforming) Data Model into Database Design

Each entity will map to a table. So we will create an EMPLOYEES Table and a DEPARTMENTS Table.

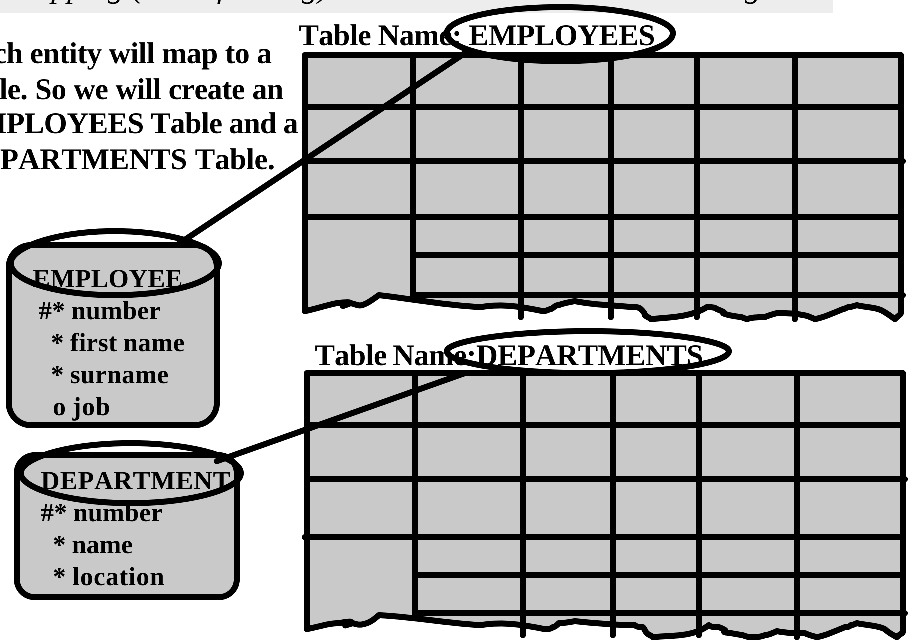


Table Name: EMPLOYEES

EMPLOYEE
#* number
*** first name**
*** surname**
o job

Table Name:DEPARTMENTS

DEPARTMENT
#* number
*** name**
*** location**

Mapping (Transforming) Data Model into Database Design

Map Attributes

Each Attribute will map to a column

E-R MODEL	**RELATIONAL DATABASE DESIGN**
Entity	**Table**
Attribute	**Column**
Unique Identifier	**Primary Key**
Relationship	**Foreign Key**

Mapping (Transforming) Data Model into Database Design

For example, the attribute EMPLOYEE number will map the column EMPNO in the EMPLOYEES Table.

EMPLOYEE
#* number
*** first name**
*** surname**
o job

DEPARTMENT
#* number
*** name**
*** location**

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	
Key Type					
Nulls/ Unique					
Sample Data					

Table Name:DEPARTMENTS

Column Name	DEPTNO	DNAME	LOC
Key Type			
Nulls/ Unique			
Sample Data			

Mapping (Transforming) Data Model into Database Design

At this point in the database design, we will add sample data to the table to provide a visual check of the table 's contents.

Those attributes labeled "*" for mandatory will be marked "NN" for NOT NULL in the table design. The RDBMS will not allow a column marked NOT NULL to contain a missing or undefined value.

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	
Key Type					
Nulls/ Unique	NN	NN	NN		
Sample Data	7369	MARY	SMITH		
	7902	HENRY	FORD	ANALYST	

Table Name:DEPARTMENTS

Column Name	DEPTNO	DNAME	LOC
Key Type			
Nulls/ Unique	NN	NN	NN
Sample Data	10	ACCOUNTING	NEWYORK
	20	RESEARCH	DALLAS

Mapping (Transforming) Data Model into Database Design

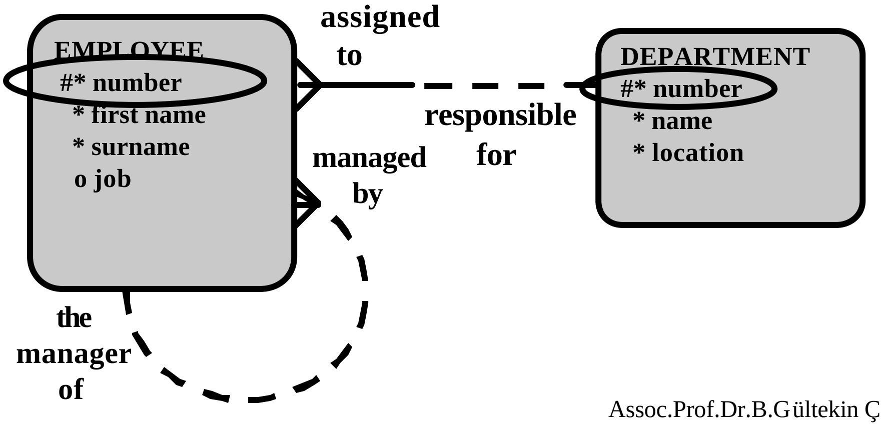
Map Unique Identifiers

Next we will map each entity’s Unique Identifier (UID) to the corresponding table ’s primary key.

E-R MODEL		RELATIONAL DATABASE DESIGN
Entity	→	**Table**
Attribute	→	**Column**
Unique Identifier	→	**Primary Key**
Relationship	→	**Foreign Key**

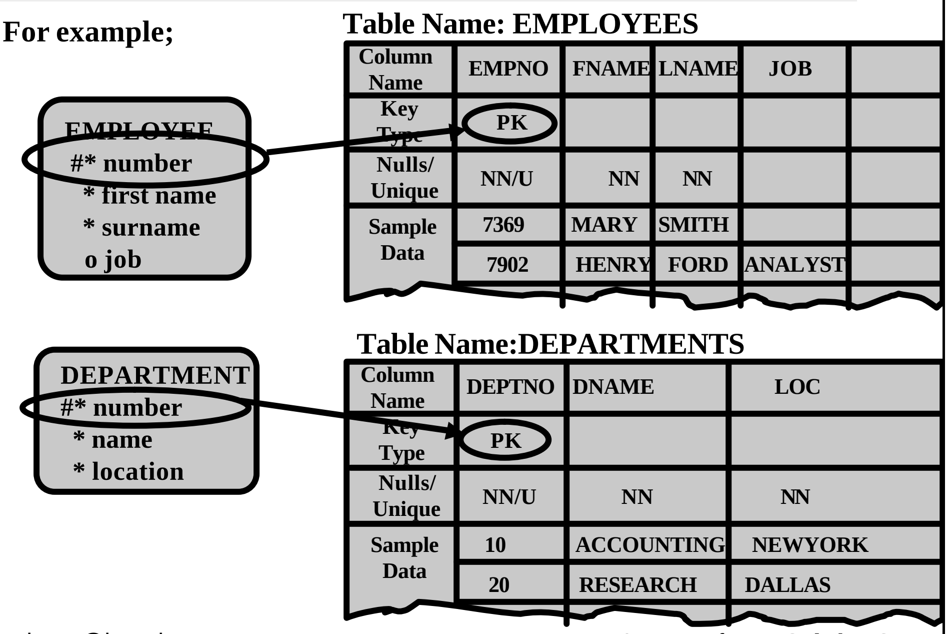
Mapping (Transforming) Data Model into Database Design

The UID of the EMPLOYEE entity is the attribute EMPLOYEE number. So, the primary key of the EMPLOYEES Table will be the column EMPNO. We will label the EMPNO column PK for Primary Key and U for Unique. Likewise for the DEPARTMENT entity, the UID attribute DEPARTMENT number will map to the primary key DEPTNO.



Mapping (Transforming) Data Model into Database Design

For example;



EMPLOYEE
#* number
*** first name**
*** surname**
o job

DEPARTMENT
#* number
*** name**
*** location**

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	
Key Type	PK				
Nulls/ Unique	NN/U	NN	NN		
Sample Data	7369	MARY	SMITH		
	7902	HENRY	FORD	ANALYST	

Table Name:DEPARTMENTS

Column Name	DEPTNO	DNAME	LOC
Key Type	PK		
Nulls/ Unique	NN/U	NN	NN
Sample Data	10	ACCOUNTING	NEWYORK
	20	RESEARCH	DALLAS

Mapping (Transforming) Data Model into Database Design

Map Relationships

Finally, we will map each relationship to a foreign key (FK) column.

E-R MODEL	RELATIONAL DATABASE DESIGN
Entity	Table
Attribute	Column
Unique Identifier	Primary Key
Relationship	Foreign Key

Mapping (Transforming) Data Model into Database Design

For the relationship between EMPLOYEES and DEPARTMENTS, we will add the foreign key column DEPTNO to the EMPLOYEES Table, and label it FK1. This is the first foreign key column in the EMPLOYEES Table.

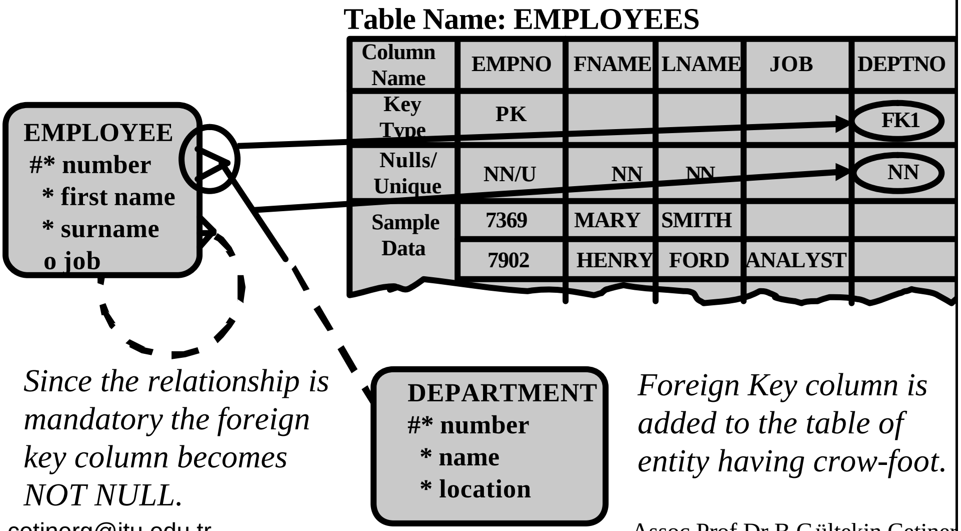


Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO
Key Type	PK				FK1
Nulls/ Unique	NN/U	NN	NN		NN
Sample Data	7369	MARY	SMITH		
	7902	HENRY	FORD	ANALYST	

Since the relationship is mandatory the foreign key column becomes NOT NULL.

Foreign Key column is added to the table of entity having crow-foot.

Mapping (Transforming) Data Model into Database Design

The foreign key DEPTNO column will allow the DEPARTMENT data for each Employee to be accessed.

For example, Mary Smith is assigned to DEPTNO=20 which is the RESEARCH department in Dallas.

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO
Key Type	PK				FK1
Nulls/ Unique	NN/U	NN	NN		NN
Sample Data	7369	MARY	SMITH		20
	7902	HENRY	FORD	ANALYST	50

Table Name:DEPARTMENTS

Column Name	DEPTNO	DNAME	LOC
Key Type			
Nulls/ Unique	NN/U	NN	NN
Sample Data	10	ACCOUNTING	NEWYORK
	20	RESEARCH	DALLAS

Mapping (Transforming) Data Model into Database Design

The recursive relationship of the EMPLOYEE entity will be mapped to a second foreign key column in the EMPLOYEES Table. We will call this column MGR and it will contain the employee number for the employee's manager.

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO	MGR
Key Type	PK				FK1	FK2
Nulls/ Unique	NN/U	NN	NN		NN	
Sample Data	7369	MARY	SMITH		20	7902
	7902	HENRY	FORD	ANALYST	50	7566

Mapping (Transforming) Data Model into Database Design

For example, Mary Smith's manager is employee number 7902-Henry Ford.

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO	MGR
Key Type	PK				FK1	FK2
Nulls/ Unique	NN/U	NN	NN		NN	
Sample Data	7369	MARY	SMITH		20	7902
	7902	HENRY	FORD	ANALYST	50	7566

Mapping (Transforming) Data Model into Database Design

Now we have a complete relational database design for the EMPLOYEES and DEPARTMENTS Tables.

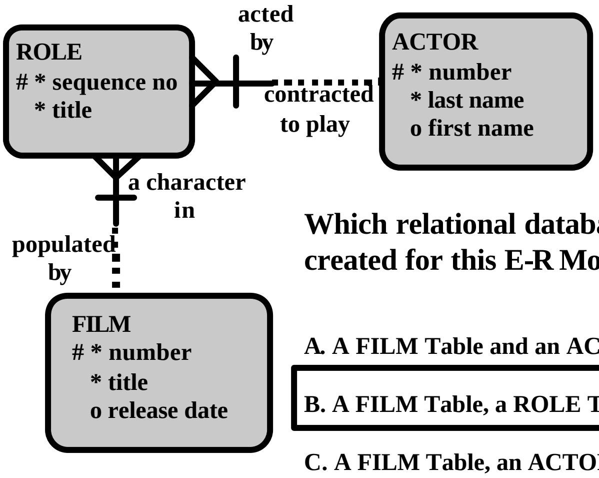
Table Name:DEPARTMENTS

Column Name	DEPTNO	DNAME	LOC
Key Type			
Nulls/ Unique	NN/U	NN	NN
Sample Data	10	ACCOUNTING	NEWYORK
	20	RESEARCH	DALLAS

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO	MGR
Key Type	PK				FK1	FK2
Nulls/ Unique	NN/U	NN	NN		NN	
Sample Data	7369	MARY	SMITH		20	7902
	7902	HENRY	FORD	ANALYST	50	7566

Mapping (Transforming) Data Model into Database Design



Which relational database tables will be created for this E-R Model?

A. A FILM Table and an ACTOR Table

[✓ B. A FILM Table, a ROLE Table and an ACTOR Table]

C. A FILM Table, an ACTOR and a CHARACTER Table

D. A FILM/ROLE Table

Mapping (Transforming) Data Model into Database Design

Which column(s) of the ACTOR Table will be the primary key?

ACTOR
# * number
*** last name**
o first name

ACTOR

ACTOR_NUM	LAST_NAME	FIRST_NAME
1557	WAYNE	JOHN
499	HURT	WILLIAM
2407	TURNER	KATHLEEN

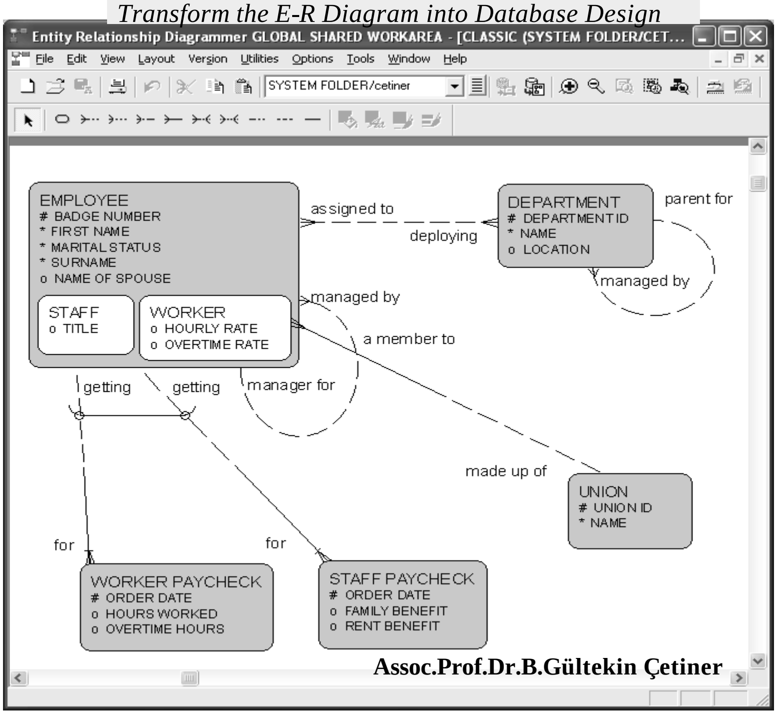
A. FIRST_NAME, LAST_NAME

B. FIRST_NAME

[✓ **C. ACTOR_NUM**]

D. ACTOR_NUM,LAST_NAME

Transform the E-R Diagram into Database Design



Assoc.Prof.Dr.B.Gültekin Çetiner

Understand Further Database Design

Further on Database Design

Referential Integrity

Referential integrity addresses database “consistency” ensuring that the values in the various tables of the database are consistent. Referential integrity deals specifically with the consistency of foreign keys.

Further on Database Design

The DEPTNO column in the EMPLOYEE Table is a foreign key column and references the primary key of the DEPARTMENT Table. The business rule states:

"Each EMPLOYEE must be assigned to one and only one DEPARTMENT".

A DEPTNO value is valid only if it references a valid DEPTNO in the DEPARTMENT Table.

Further on Database Design

You cannot enter a DEPTNO value in EMPLOYEES Table if that is not available in DEPARTMENTS Table.

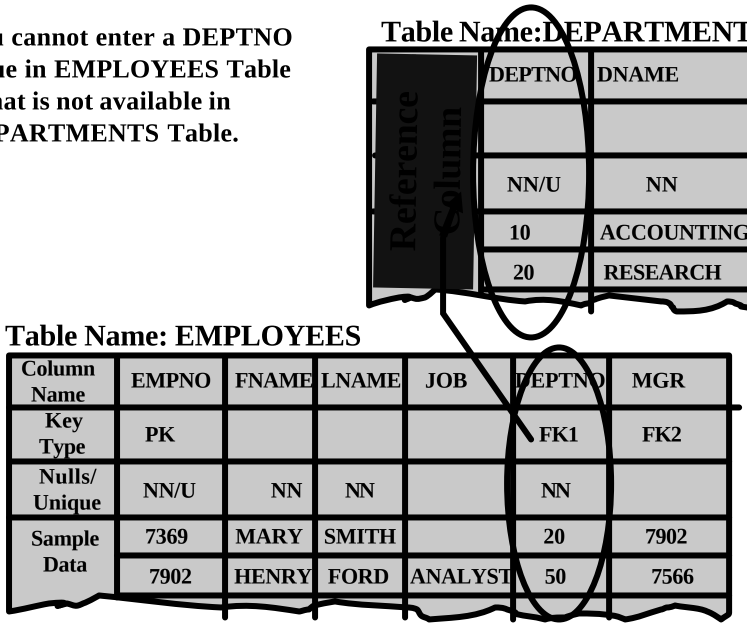
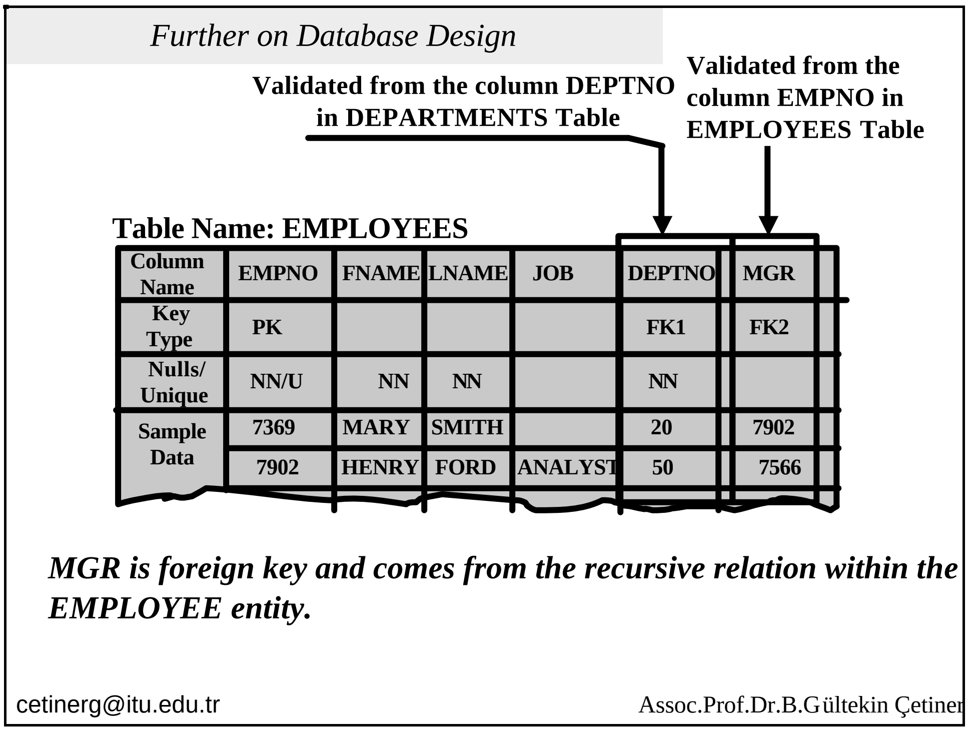


Table Name:DEPARTMENTS

DEPTNO	DNAME	LOC
NN/U	NN	NN
10	ACCOUNTING	NEWYORK
20	RESEARCH	DALLAS

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO	MGR
Key Type	PK				FK1	FK2
Nulls/ Unique	NN/U	NN	NN		NN	
Sample Data	7369	MARY	SMITH		20	7902
	7902	HENRY	FORD	ANALYST	50	7566



Further on Database Design

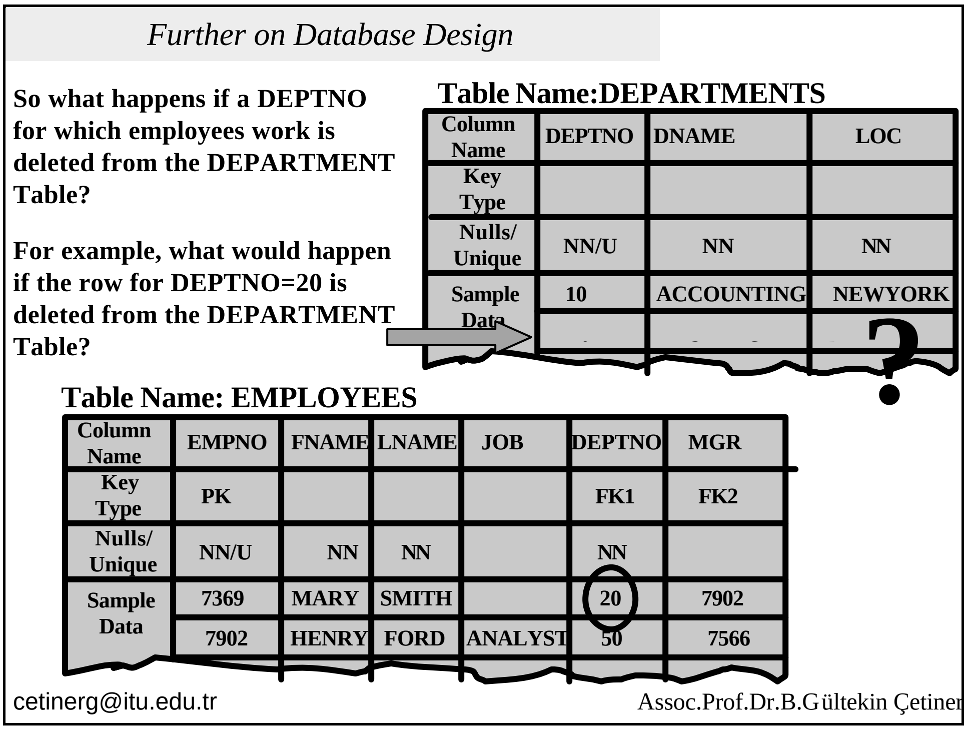
Validated from the column DEPTNO in DEPARTMENTS Table

Validated from the column EMPNO in EMPLOYEES Table

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO	MGR
Key Type	PK				FK1	FK2
Nulls/ Unique	NN/U	NN	NN		NN	
Sample Data	7369	MARY	SMITH		20	7902
	7902	HENRY	FORD	ANALYST	50	7566

MGR is foreign key and comes from the recursive relation within the EMPLOYEE entity.



Further on Database Design

So what happens if a DEPTNO for which employees work is deleted from the DEPARTMENT Table?

For example, what would happen if the row for DEPTNO=20 is deleted from the DEPARTMENT Table?

Table Name:DEPARTMENTS

Column Name	DEPTNO	DNAME	LOC
Key Type			
Nulls/ Unique	NN/U	NN	NN
Sample Data	10	ACCOUNTING	NEWYORK

?

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO	MGR
Key Type	PK				FK1	FK2
Nulls/ Unique	NN/U	NN	NN		NN	
Sample Data	7369	MARY	SMITH		20	7902
	7902	HENRY	FORD	ANALYST	50	7566

Further on Database Design

What happens depends on what referential integrity rule was specified for the FK DEPTNO in the EMPLOYEES Table. The database designer or DBA should specify a referential integrity rule for every foreign key in the database.

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO	MGR
Key Type	PK				FK1	FK2
Nulls/ Unique	NN/U	NN	NN		NN	
Sample Data	7369	MARY	SMITH		20	7902
	7902	HENRY	FORD	ANALYST	50	7566

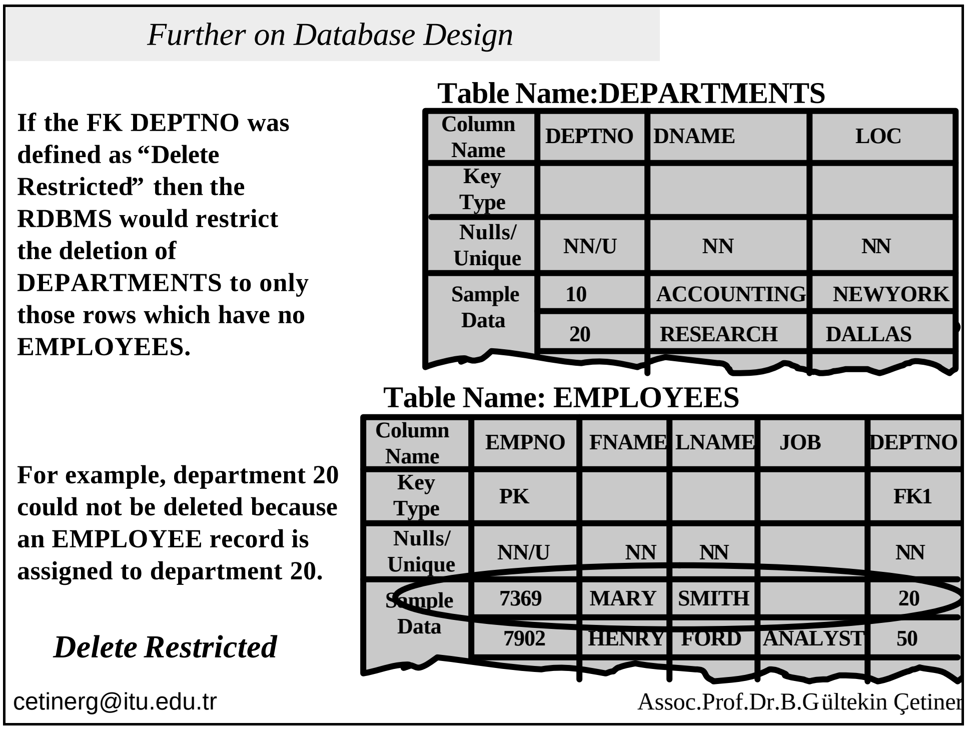
Further on Database Design

The database designer can specify one of three options:

<u>Delete restricted</u>, which restricts the deletion of certain rows in the table.

<u>Delete cascade</u>, which deletes the corresponding rows of the associated table.

<u>Delete nullify</u>, which places null values in the corresponding rows of the associated table.



Further on Database Design

If the FK DEPTNO was defined as "Delete Restricted" then the RDBMS would restrict the deletion of DEPARTMENTS to only those rows which have no EMPLOYEES.

For example, department 20 could not be deleted because an EMPLOYEE record is assigned to department 20.

Delete Restricted

Table Name:DEPARTMENTS

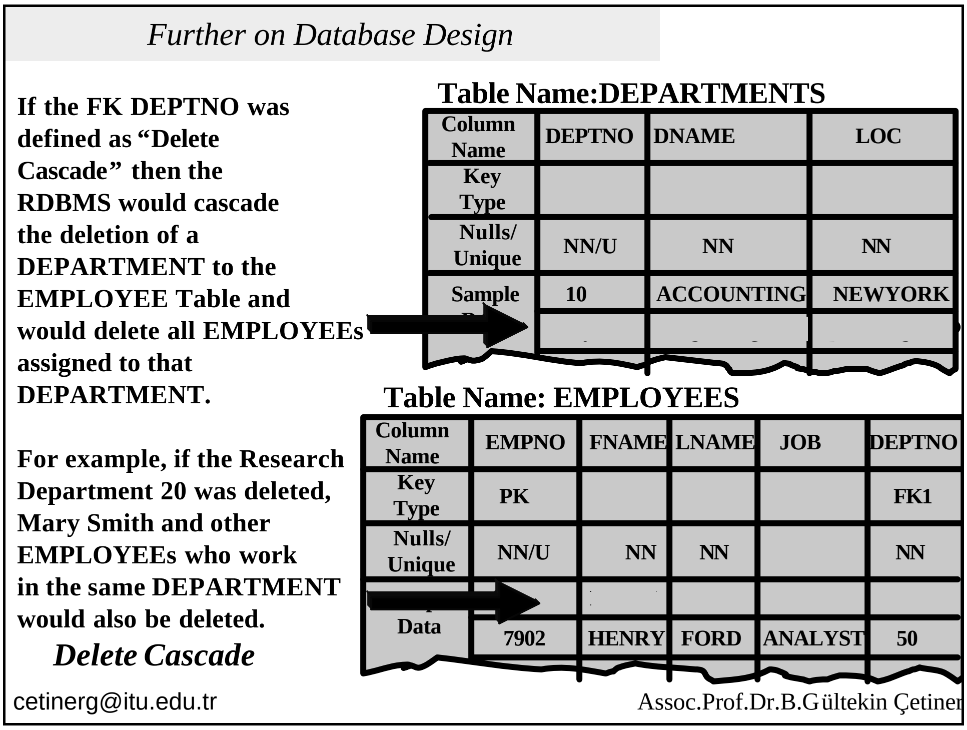
Column Name	DEPTNO	DNAME	LOC
Key Type			
Nulls/ Unique	NN/U	NN	NN
Sample Data	10	ACCOUNTING	NEWYORK
	20	RESEARCH	DALLAS

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO
Key Type	PK				FK1
Nulls/ Unique	NN/U	NN	NN		NN
Sample Data	7369	MARY	SMITH		20
	7902	HENRY	FORD	ANALYST	50

cetinerg@itu.edu.tr

Assoc.Prof.Dr.B.Gültekin Çetiner



Further on Database Design

If the FK DEPTNO was defined as "Delete Cascade" then the RDBMS would cascade the deletion of a DEPARTMENT to the EMPLOYEE Table and would delete all EMPLOYEEs assigned to that DEPARTMENT.

For example, if the Research Department 20 was deleted, Mary Smith and other EMPLOYEEs who work in the same DEPARTMENT would also be deleted.

Delete Cascade

Table Name:DEPARTMENTS

Column Name	DEPTNO	DNAME	LOC
Key Type			
Nulls/ Unique	NN/U	NN	NN
Sample	10	ACCOUNTING	NEWYORK

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO
Key Type	PK				FK1
Nulls/ Unique	NN/U	NN	NN		NN
Data	7902	HENRY	FORD	ANALYST	50

cetinerg@itu.edu.tr

Assoc.Prof.Dr.B.Gültekin Çetiner

Further on Database Design

If employee assignment to a DEPARTMENT was optional and the FK DEPTNO was defined as "Delete Nullify" the RDBMS would nullify any references to a department when that department was deleted.

Delete Nullify

Table Name:DEPARTMENTS

Column Name	DEPTNO	DNAME	LOC
Key Type			
Nulls/ Unique	NN/U	NN	NN
Sample	10	ACCOUNTING	NEWYORK

Table Name: EMPLOYEES

Column Name	EMPNO	FNAME	LNAME	JOB	DEPTNO
Key Type	PK				FK1
Nulls/ Unique	NN/U	NN	NN		NN
Data	7369	MARY	SMITH		
	7902	HENRY	FORD	ANALYST	50

Further on Database Design

Referential integrity rules should be assigned to all foreign keys in a database

Further on Database Design

You can assign one of the three rules for each foreign key.

Further on Database Design

Indexing

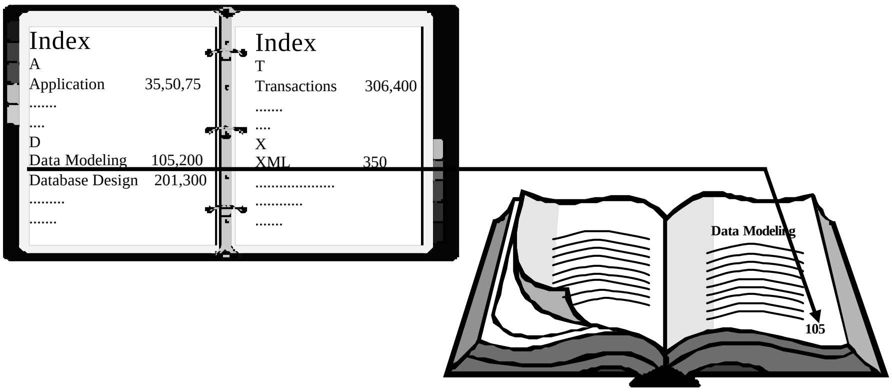
Once the referential integrity constraints have been identified, it is time to decide on the indexing of the tables.

Indexes provide the DBMS with a quick way of looking up the location of rows, rather than sequentially scanning the table to satisfy every request.

Further on Database Design

Indexing

Indexing is like using indexes on a book to find certain items by looking at the certain things (e.g. Keywords)

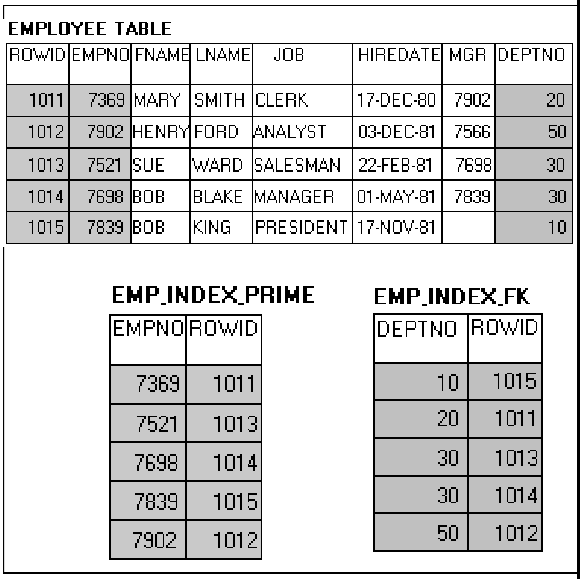


Further on Database Design

Most DBMSs require that primary key be indexed, in order to enforce the requirement that a primary key must be unique. Most DBMSs do not require that foreign keys be indexed, but all of them normally would be indexed.

Further on Database Design

For example, for the EMPLOYEES Table, a unique index might be created on the PK column EMPNO and a non-unique index might be created on the FK column DEPTNO.



EMPLOYEE TABLE

ROWID	EMPNO	FNAME	LNAME	JOB	HIREDATE	MGR	DEPTNO
1011	7369	MARY	SMITH	CLERK	17-DEC-80	7902	20
1012	7902	HENRY	FORD	ANALYST	03-DEC-81	7566	50
1013	7521	SUE	WARD	SALESMAN	22-FEB-81	7698	30
1014	7698	BOB	BLAKE	MANAGER	01-MAY-81	7839	30
1015	7839	BOB	KING	PRESIDENT	17-NOV-81		10

EMP_INDEX_PRIME

EMPNO	ROWID
7369	1011
7521	1013
7698	1014
7839	1015
7902	1012

EMP_INDEX_FK

DEPTNO	ROWID
10	1015
20	1011
30	1013
30	1014
50	1012

Further on Database Design

In addition, we will index any column or combination that is frequently used as a search key (i.e., in an SQL WHERE clause), or as a sort key (i.e., an SQL ORDER BY, GROUP BY, or similar clause).

Further on Database Design

A table will usually have two to four indexes to support a typical mixture of transaction processing, batch jobs, and ad hoc query.

The greater the proportion of read-only processing, the more beneficial indexes will be.

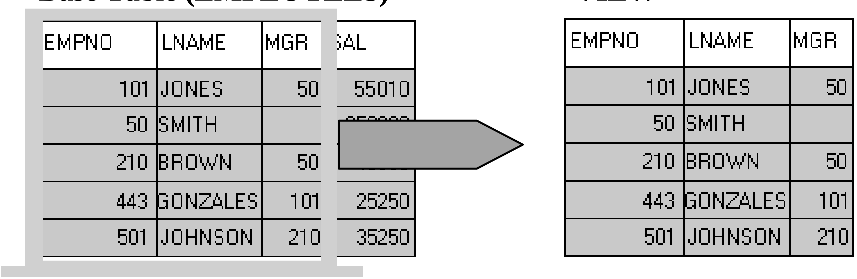
Further on Database Design

The greater the proportion of update processing, the less beneficial indexes will be. For example, when a row is inserted into a table, the proper keys must be inserted into each index. If a table had 12 indexes, the insertion of a new row would be the equivalent of updating 13 files.

Further on Database Design

Views

A view is like a "window" onto a table - a window which can reveal only certain columns and/or rows, or can change the appearance of the data. A view of the EMPLOYEES Table could be used to restrict users from seeing employees' salaries.



Base Table (EMPLOYEES)

EMPNO	LNAME	MGR	SAL
101	JONES	50	55010
50	SMITH		[illegible]
210	BROWN	50	[illegible]
443	GONZALES	101	25250
501	JOHNSON	210	35250

VIEW

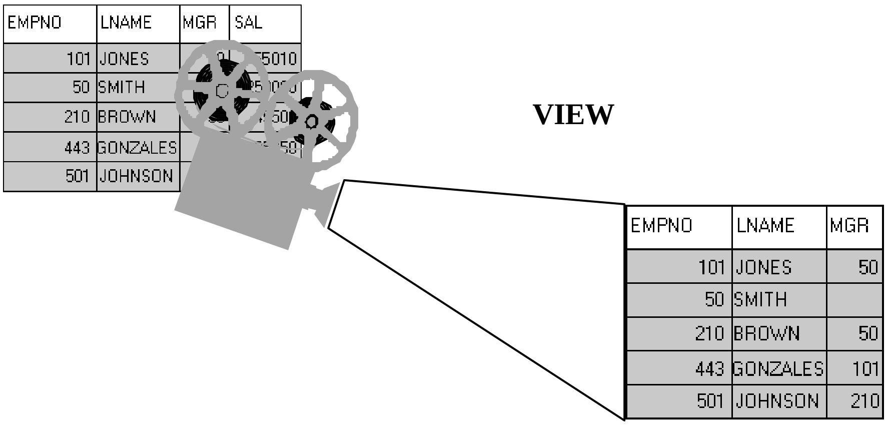
EMPNO	LNAME	MGR
101	JONES	50
50	SMITH	
210	BROWN	50
443	GONZALES	101
501	JOHNSON	210

Further on Database Design

Views

A view is also like a projector which reflects the base table onto a surface with a required form.

Base Table (EMPLOYEES)



EMPNO	LNAME	MGR	SAL
101	JONES	[illegible]	55010
50	SMITH	[illegible]	[illegible]
210	BROWN	[illegible]	[illegible]
443	GONZALES	[illegible]	[illegible]
501	JOHNSON	[illegible]	[illegible]

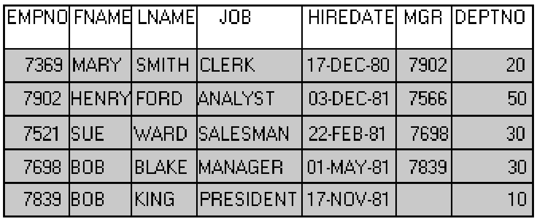
VIEW

EMPNO	LNAME	MGR
101	JONES	50
50	SMITH	
210	BROWN	50
443	GONZALES	101
501	JOHNSON	210

Further on Database Design

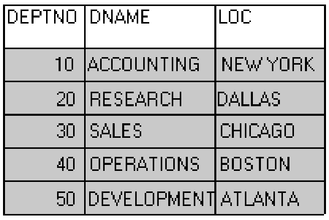
A view can also be used to present normalized data in a denormalized form. For example, following the rules of normalization, the EMPLOYEE and DEPARTMENT Tables are separate.

Table Name: ***EMPLOYEES***



EMPNO	FNAME	LNAME	JOB	HIREDATE	MGR	DEPTNO
7369	MARY	SMITH	CLERK	17-DEC-80	7902	20
7902	HENRY	FORD	ANALYST	03-DEC-81	7566	50
7521	SUE	WARD	SALESMAN	22-FEB-81	7698	30
7698	BOB	BLAKE	MANAGER	01-MAY-81	7839	30
7839	BOB	KING	PRESIDENT	17-NOV-81		10

Table Name: ***DEPARTMENTS***

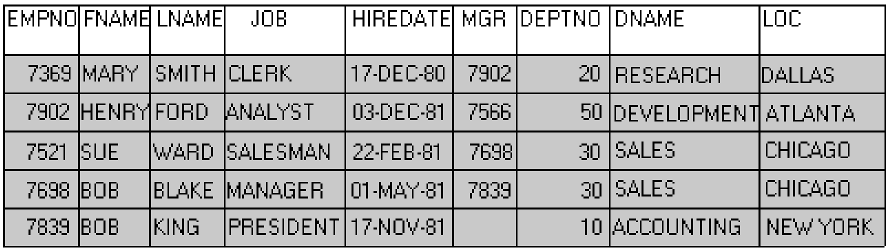


DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON
50	DEVELOPMENT	ATLANTA

Further on Database Design

A view defined accross both tables could be used to prejoin the tables so the user would only see a single "logical tale".

VIEW_EMPLOYEES



EMPNO	FNAME	LNAME	JOB	HIREDATE	MGR	DEPTNO	DNAME	LOC
7369	MARY	SMITH	CLERK	17-DEC-80	7902	20	RESEARCH	DALLAS
7902	HENRY	FORD	ANALYST	03-DEC-81	7566	50	DEVELOPMENT	ATLANTA
7521	SUE	WARD	SALESMAN	22-FEB-81	7698	30	SALES	CHICAGO
7698	BOB	BLAKE	MANAGER	01-MAY-81	7839	30	SALES	CHICAGO
7839	BOB	KING	PRESIDENT	17-NOV-81		10	ACCOUNTING	NEW YORK

Further on Database Design

The view is simply a SQL SELECT statement stored in the DBMS’s catalog. Any SELECT statement, with a few exceptions imposed by the various DBMSs, can serve as the definition of a view.

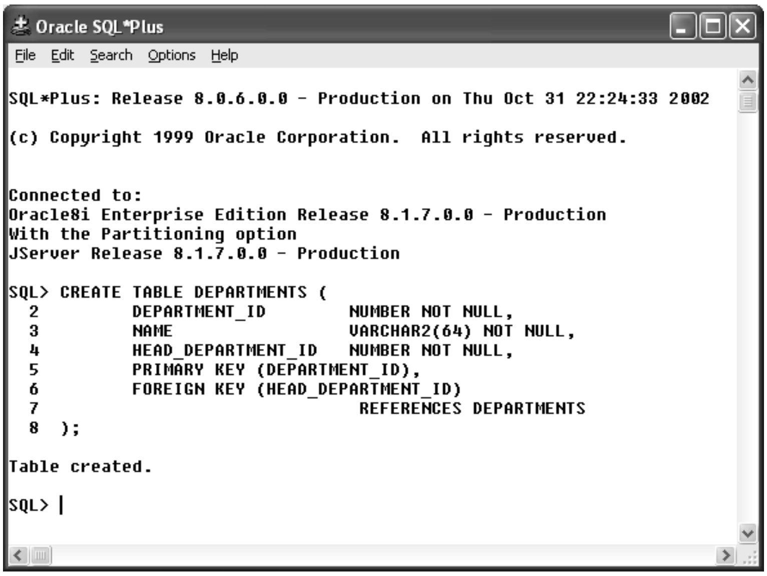
Further on Database Design

Database Build

After addressing referential integrity, designing indexes, and creating views, you are now ready to create real RDBMS tables using the SQL CREATE statement. For example, to create the physical tables for the DEPARTMENTS and EMPLOYEES tables, you must use the following SQL statements.

Further on Database Design

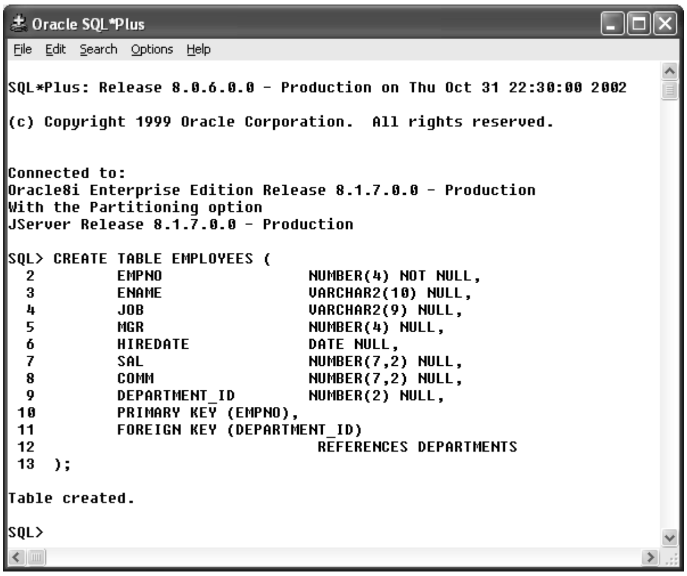
For DEPARTMENTS Table;



```
SQL*Plus: Release 8.0.6.0.0 - Production on Thu Oct 31 22:24:33 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Connected to:
Oracle8i Enterprise Edition Release 8.1.7.0.0 - Production
With the Partitioning option
JServer Release 8.1.7.0.0 - Production

SQL> CREATE TABLE DEPARTMENTS (
  2         DEPARTMENT_ID        NUMBER NOT NULL,
  3         NAME                 VARCHAR2(64) NOT NULL,
  4         HEAD_DEPARTMENT_ID   NUMBER NOT NULL,
  5         PRIMARY KEY (DEPARTMENT_ID),
  6         FOREIGN KEY (HEAD_DEPARTMENT_ID)
  7                               REFERENCES DEPARTMENTS
  8  );

Table created.

SQL> |
```

Further on Database Design

For EMPLOYEES Table;



```
SQL*Plus: Release 8.0.6.0.0 - Production on Thu Oct 31 22:30:00 2002

(c) Copyright 1999 Oracle Corporation.  All rights reserved.


Connected to:
Oracle8i Enterprise Edition Release 8.1.7.0.0 - Production
With the Partitioning option
JServer Release 8.1.7.0.0 - Production

SQL> CREATE TABLE EMPLOYEES (
  2         EMPNO                NUMBER(4) NOT NULL,
  3         ENAME                VARCHAR2(10) NULL,
  4         JOB                  VARCHAR2(9) NULL,
  5         MGR                  NUMBER(4) NULL,
  6         HIREDATE             DATE NULL,
  7         SAL                  NUMBER(7,2) NULL,
  8         COMM                 NUMBER(7,2) NULL,
  9         DEPARTMENT_ID        NUMBER(2) NULL,
 10         PRIMARY KEY (EMPNO),
 11         FOREIGN KEY (DEPARTMENT_ID)
 12                               REFERENCES DEPARTMENTS
 13  );

Table created.

SQL>
```