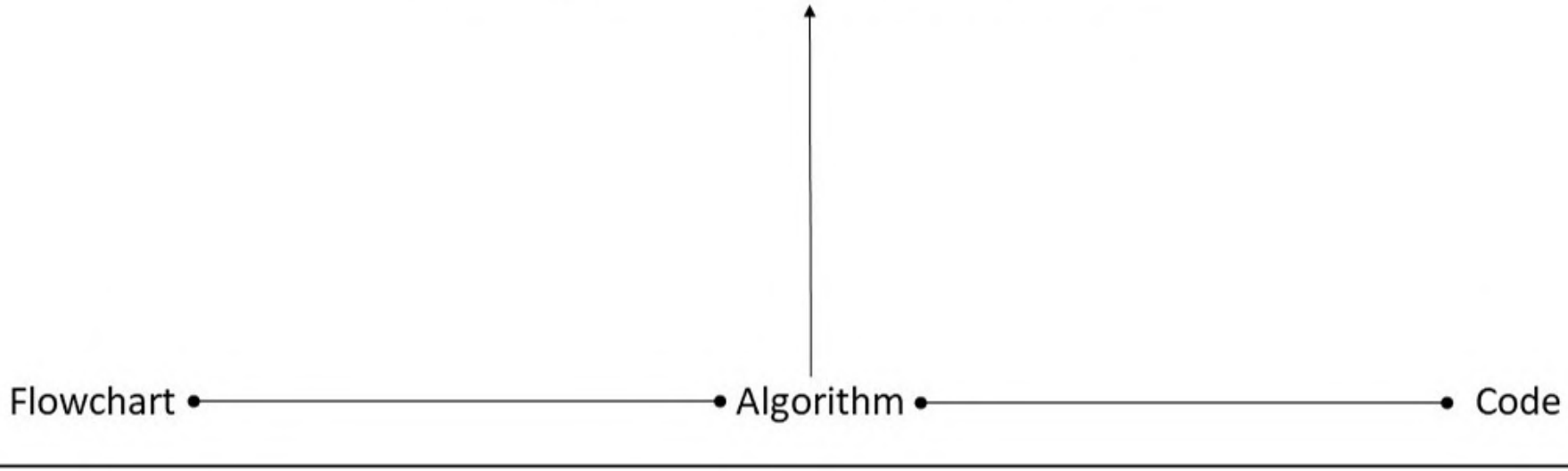
Relational Model

From the Realm of Rectangles and Rhombuses to the World of Rows and Columns

Why **Relational** Model



Chapter Organization

- Intro & Terminologies
- RM formally defined (Set Theory basis)
- Components of Relational Model
- Constraints on RM
  - Basic Constraints
  - Key
  - Foreign Key

What is a Relational Model ?

- It is a conceptual model/tool which uses collection of “Tables” to represent data & its Inter-relationships.

- Proposed by Edgar F. Codd (Ted Codd) in the 1970s

Its not a Table actually, But it “looks like” a table

RM is based upon Set Theory and First Order Predicate Logic

Relational Model, allows data to be organized in such a way that **SQL** queries can be easily executed on them.

RM: In Simple Terms

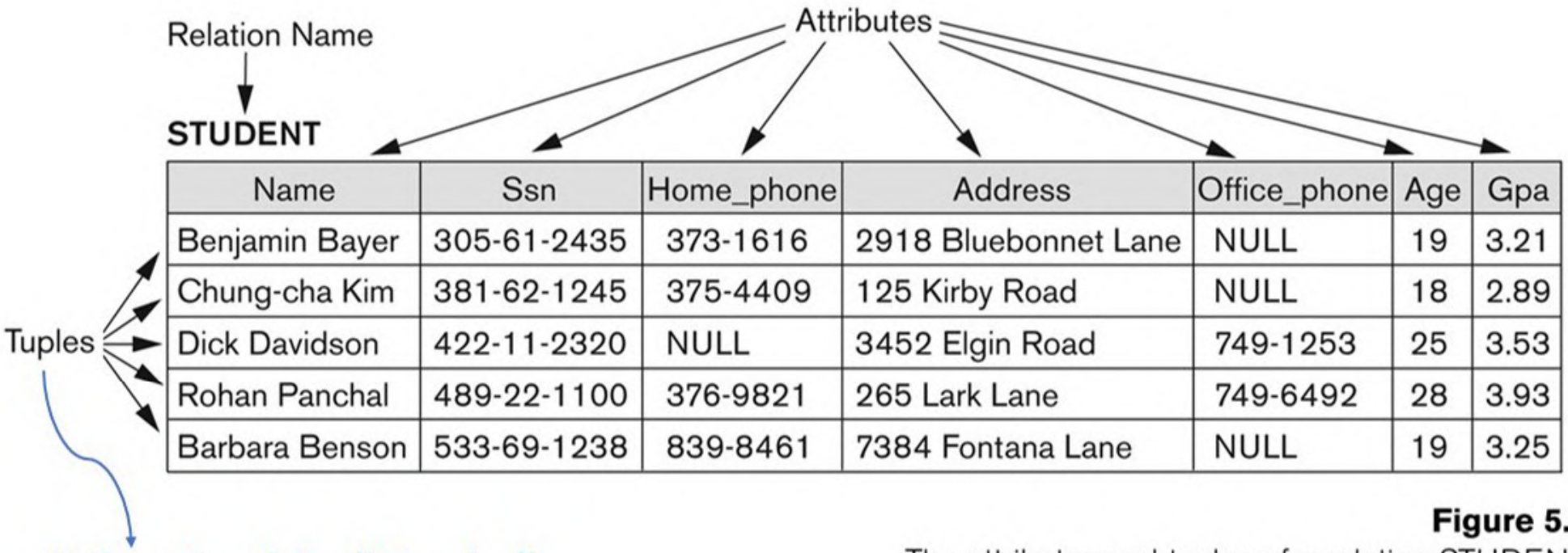
Everything is a Table here

The entire DB is a Collection of Tables

In RM, everything is a SET, ➔ Set operations can be easily performed upon them.

Relation ⟷ More or Less Same Meaning ⟷ Table

Informally A Relation is a Table of Values (Rows)



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

Figure 5.1
The attributes and tuples of a relation STUDENT.

What is a Tuple ?

Simple Definition: Tuples are Just "**Rows**" in a table

Formal Definition: Tuple is an ordered sequence of values $<v_1, v_2, ..v_i, ..v_n>$

$t[a_i]$ refers to the value of v_i corresponding to attribute a_i

It is the value of a_i in the t^{th} row

Tuple ⟷ More or Less / Same Meaning ⟷ Row

Example:

STUDENT

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

u[Ssn] = 422-11-2320

t[Age,Gpa] = 19, 3.25

As an example, consider the tuple t = <'Barbara Benson', '533-69-1238', '(817)839-8461', '7384 Fontana Lane', NULL, 19, 3.25> from the STUDENT relation in Figure 3.1; we have t[Name] = <'Barbara Benson'>, and t[Ssn, Gpa, Age] = <'533-69-1238', 3.25, 19>.

Example:

STUDENT

	Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
t_1	Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
t_2	Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25
t_3	Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
.	Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
.	Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21

$t_1[Ssn] = 422\text{-}11\text{-}2320$

$t_i[Ssn] \neq t_j[Ssn]$ for all i and j; $i \neq j$

➔ Ssn is a Key

What is an Attribute

Its just the name of the column, which helps to record information

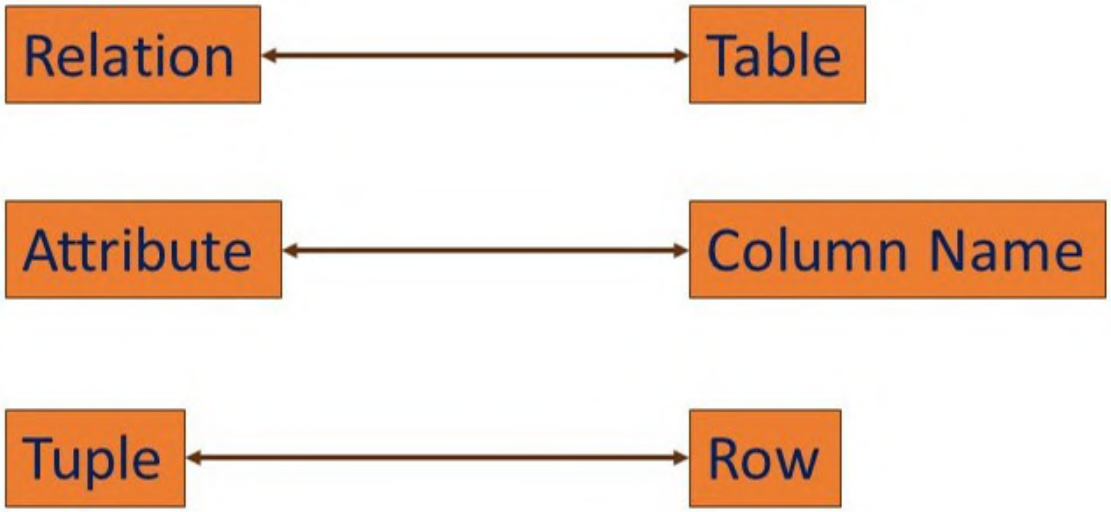
Every attribute has a "**Domain**", which is the set of possible values that attribute can take.

As per RM, Domain should be a set of Atomic Values only

➔ No Composite or Multivalued attributes are allowed

Attribute ⟷ More or Less Same Meaning ⟷ Column Name

RM Terminologies



Relational Database (**RDB**) – A DB that is organized based on the RM.

RDB is simply a collection of "Named Relations"

Chapter Organization

- Intro & Terminologies
- **RM formally defined (Set Theory basis)**
- Components of Relational Model
- Constraints on RM
  - Basic Constraints
  - Key
  - Foreign Key

RM: In Formal Terms

RM Represents data as a collection of Relations

What is a Relation ??. Lets discuss that later.
As of now Relation is just a 'table' of values to us

Rows / Tuples represents facts related to real world entity.

The table name and the Column names (Attributes) helps to interpret the meaning of each value in a row

ER vs Relational Model

High Level Conceptual / Pictorial Description of the DB

For Users / Common People

Doesn't worry about retrieval / No SQL

Primary focus is on Ideation & design

Abstract Representation of Data

Logical description of the DB

For Programmers / DBA

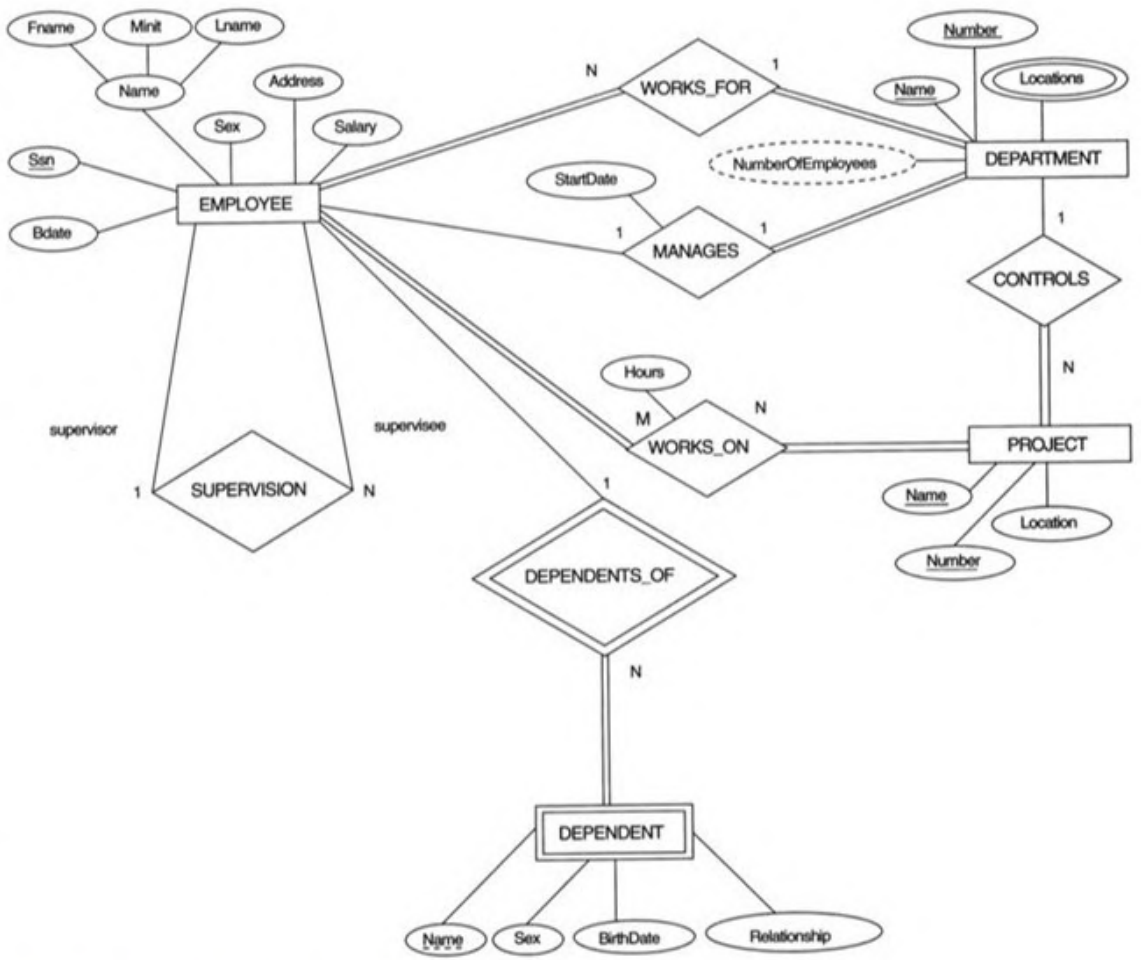
Organized for Efficient SQL Execution

Focus is on storage, query execution, and manipulation

Concrete, Implementable representation of Data

From Conceptual Diagrams to Concrete Tables

GOAL:
From This



To This

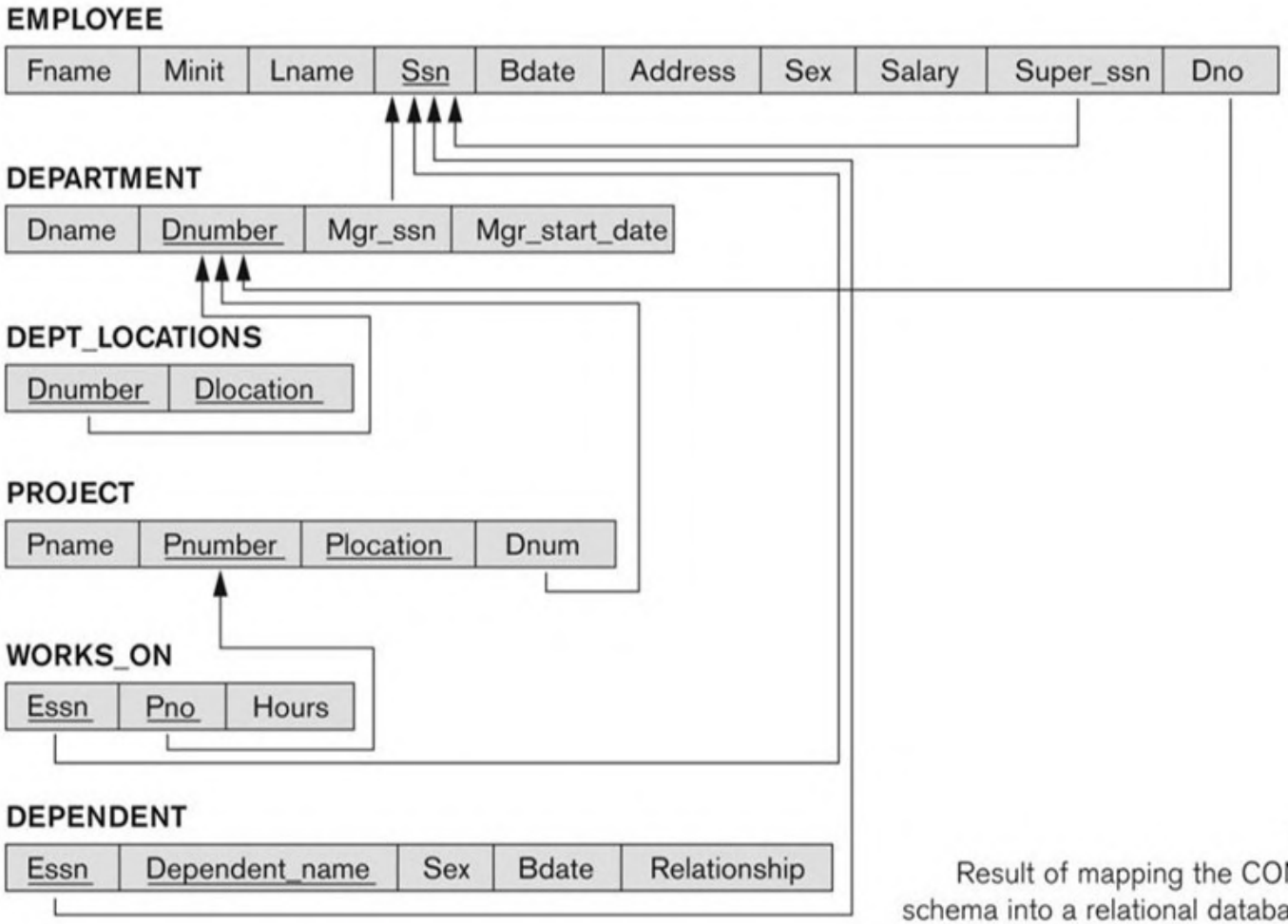


Figure 7.2
Result of mapping the COMPANY ER schema into a relational database schema.

Storing Data as a Table, Isn't that Obvious, Should I even study this ?

Relational model may look simple and obvious, but before its invention DB design was quite complex.

We are so used to the notion of storing data as tables (Excel) that this looks obvious.

Edgar F Codd, was given Turing award for this.

Advantages of Adopting Relational Model:

It is Simple and easy to understand.
In fact in some cases people skip, ER modelling and go straight to the RM.

Visualising data as a table is both intuitive to the developer and as well is concrete enough to be implemented.

Relational Model led to the development of Data Dependencies & Normalization.

Huge Improvement in bringing down Redundancy & Inconsistency

The “Relation” in Relational Model

Do you remember the definition of **Relation** from Set Theory ?

A relation is a subset of a cartesian product

Example: Less Than (<) Relation

Let's define two sets:

- **Set A** = {1, 2, 3}
- **Set B** = {2, 3, 4}

The **Cartesian product** A×B contains all possible pairs:

$A \times B = \{(1,2),(1,3),(1,4),(2,2),(2,3),(2,4),(3,2),(3,3),(3,4)\}$

Now, let's define a **relation** R:

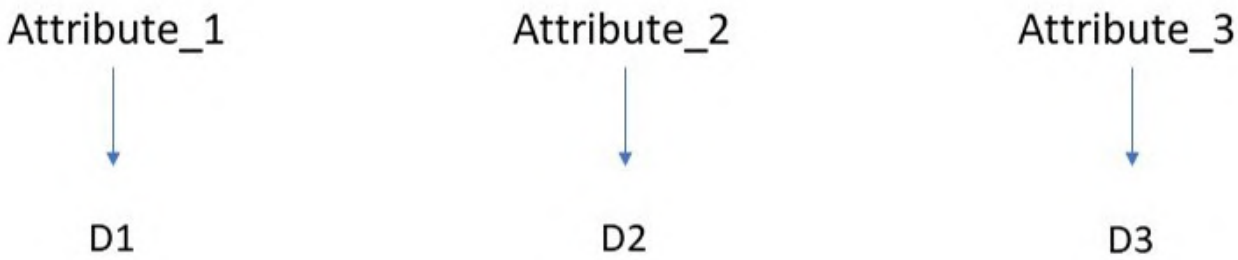
"An element from A is less than an element from B."

So, R will be:

$R = \{(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)\}$

The “Relation” in Relational Model:

Consider an Entity with few Attributes and its Corresponding Domains



Relation $\subset$ D1 X D2 X D3

The earlier definition of a relation can be *restated* more formally using set theory concepts as follows. A relation (or relation state) $r(R)$ is a **mathematical relation** of degree n on the domains $\text{dom}(A_1)$, $\text{dom}(A_2)$, ..., $\text{dom}(A_n)$, which is a **subset** of the **Cartesian product** (denoted by $\times$) of the domains that define R:

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

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

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

This product of cardinalities of all domains represents the total number of possible instances or tuples that can ever exist in any relation state $r(R)$. Of all these possible combinations, a relation state at a given time—the **current relation state**—reflects only the valid tuples that represent a particular state of the real world. In general, as the state of the real world changes, so does the relation state, by being transformed into another relation state. However, the schema R is relatively static and changes *very* infrequently—for example, as a result of adding an attribute to represent new information that was not originally stored in the relation.

So, What is a Relation In DBMS

It is an unordered set that contains the relationship of Attributes which represents the entities.

BOOK

ISBN	Name	Auth
1234	PS-1	Kalki
5678	Dune	Frank Herbert

This relation shows the valid "Relationship" that exists between the author, book name and ISBN, for the book entity

Chapter Organization

- Intro & Terminologies
- RM formally defined (Set Theory basis)
- **Components of Relational Model**
- Constraints on RM
  - Basic Constraints
  - Key
  - Foreign Key

Relational Model – What it is made up of ?

It Consists of:

- Relational Schema
- Relational Algebra
- Relational Calculus

1. What is a Relational Schema

A **relational schema** is the blueprint or structure that defines the organization of a relational database.

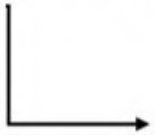
It specifies the **name of the relation (table)**, its **attributes (columns)**, and the **domain (type of values)** that each attribute can hold.

It may also include constraints, such as primary keys, foreign keys…

Simply: It's a description of the Table, without the data or rows

Relational Schema - Example

Student (Student_ID: INT, Name: CHAR(50), Age: INT, Department: CHAR(50))



This is the "Schema" of the Student table.

Information Conveyed by The Schema:

Relation Name: Student (i.e., table name)
Attributes: Student_ID, Name, Age, Dept

Domain of each attributes:
Student_Id: Integer
Name: It's a string of 50 Characters

Relational Schema - Example

Student (Student_ID: INT, Name: CHAR(50), Age: INT, Department: CHAR(50))

The Data which is going to populate this Student table with this particular schema, should conform to this specified format.

(501, Norman, 18, CSE) → Valid Row / Data

(CS20D1009, Light Yagami, 19, AI) → InValid Row, Violates the specified Schema

Schema Vs Type Declaration

Student (Student_ID: INT, Name: CHAR(50), Age: INT, Department: CHAR(50))

```
struct Student {
    int Student_ID;
    char Name[50];
    int Age;
    char Department [50]
};
```

Both are Similar Isn't it

Relational Schema Vs DB Schema

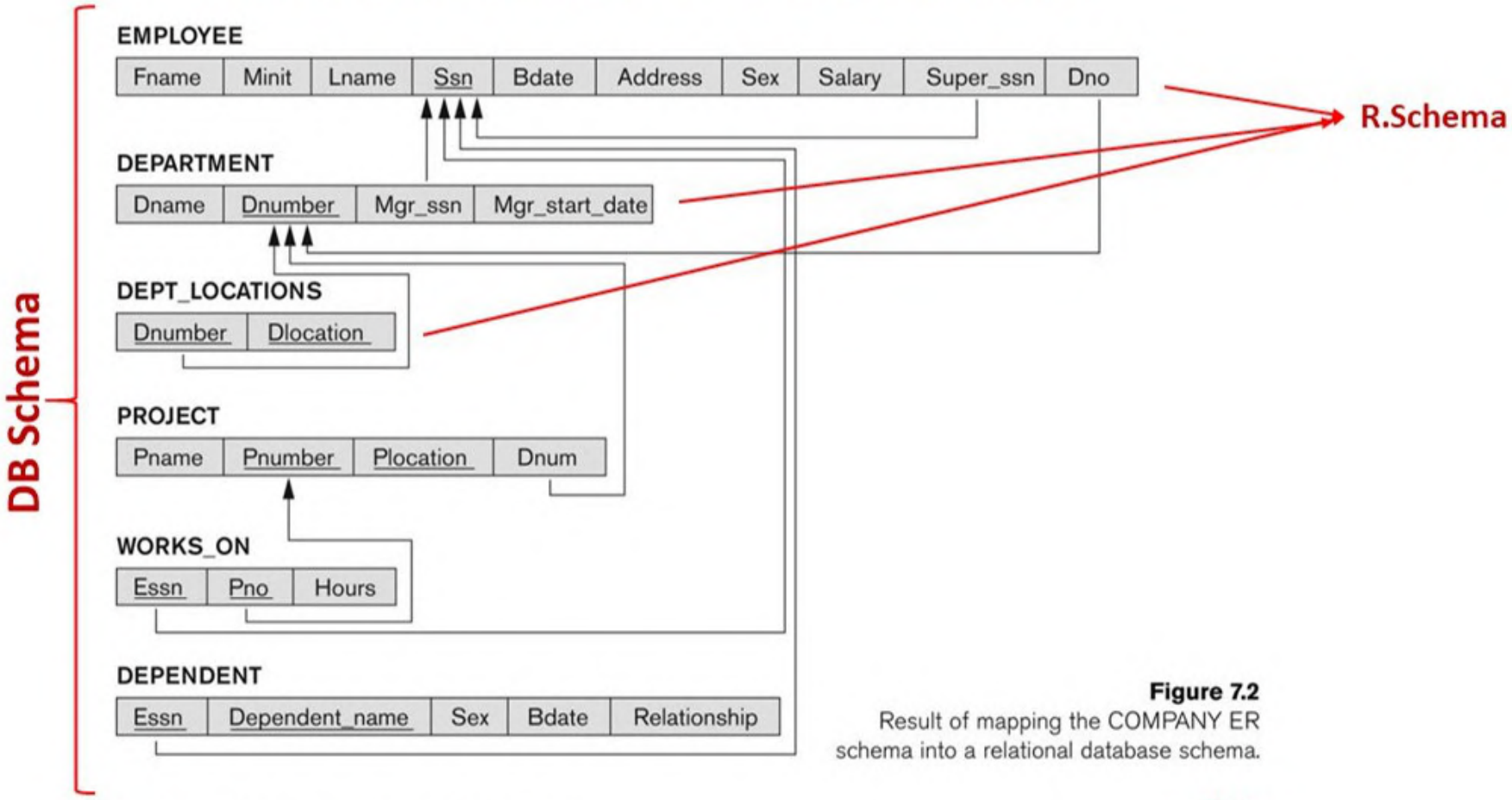


Figure 7.2
Result of mapping the COMPANY ER schema into a relational database schema.

Slide

Relational Schema Vs Relation

Relational Schema

R.Schma is denoted as $\mathbf{R\ (A_1,\ldots,A_i,\ldots,A_N)}$ N → Degree of the Relation

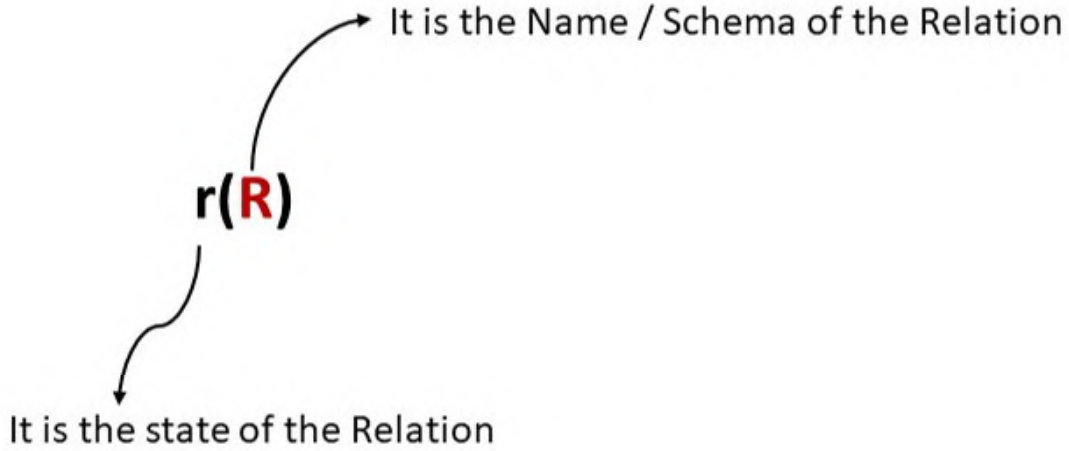
where, For all i, A_i € D_i

Relation

Denoted by '**r(R)**' ⟶ 'r' is of the schema $R(A_1,\ldots,A_i,\ldots,A_N)$

A Relation should always follow or Correspond to a R.Schema

$$R(A_1, \ldots, A_i, \ldots, A_N)$$



R.Schema	Relation
Book_Scheme(ISBN, Title, Author,...)	Book_in_First_floor(Book_Scheme) Book_in_Second_floor(Book_Scheme)
Often stays static. Rarely changes.	It may change often (i.e. contents may change) It reflects real world state.
Intension	Extension, **Relation State** Relation Instance

This is R.Schema

STUDENT

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa

STUDENT

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

One Possible Relation State

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

Another Possible Relation State

This is a DB 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 3.5
Schema diagram for the COMPANY relational database schema.

This is a DB Instance / State

State of The DB:
State of all Individual relation at a particular point in time.

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

Relational Model – Formal Definition

It Consists of:

- Relational Schema
- Relational Algebra
- Relational Calculus

Time to Define these two

What is a Relational Algebra

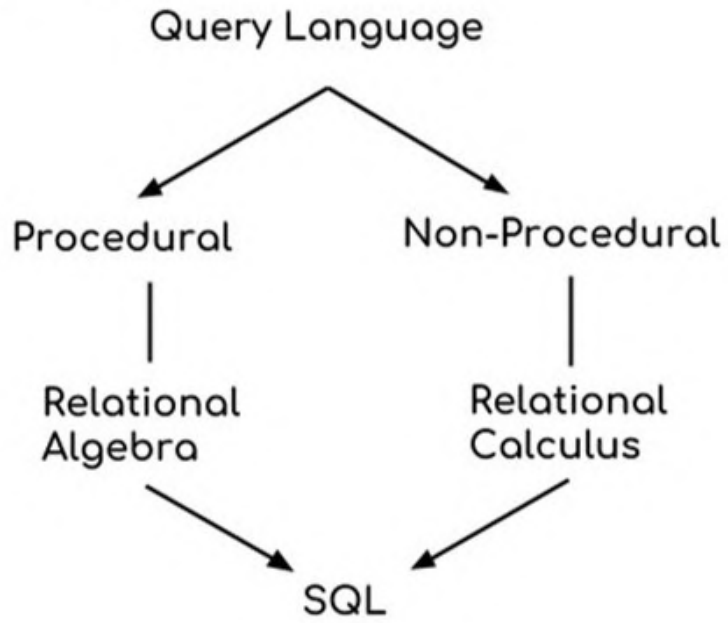
Its a technique which allows us to formally retrieve data from a Database.

It's a high level strategy – We don't specify low level operations. Simply Mention the steps needed to be taken to get the answer.

What is a Relational Calculus

Relational Calculus is a non-procedural query language that specifies what to retrieve rather than how to retrieve it.

It is a Non Procedural Language.



Important Notations

- The uppercase letters Q, R, S denote relation names.
- The lowercase letters q, r, s denote relation states.
- The letters t, u, v denote tuples.
- In general, the name of a relation schema such as STUDENT also indicates the current set of tuples in that relation—the *current relation state*—whereas STUDENT(Name, Ssn, ...) refers *only* to the relation schema.
- An attribute A can be qualified with the relation name R to which it belongs by using the dot notation $R.A$—for example, STUDENT.Name or STUDENT.Age. This is because the same name may be used for two attributes in different relations. However, all attribute names *in a particular relation* must be distinct.
- An n-tuple t in a relation $r(R)$ is denoted by $t = <v_1, v_2, ..., v_n>$, where v_i is the value corresponding to attribute A_i. The following notation refers to **component values** of tuples:
- Both $t[A_i]$ and $t.A_i$ (and sometimes $t[i]$) refer to the value v_i in t for attribute A_i.
- Both $t[A_u, A_w, ..., A_z]$ and $t.(A_u, A_w, ..., A_z)$, where $A_u, A_w, ..., A_z$ is a list of attributes from R, refer to the subtuple of values $<v_u, v_w, ..., v_z>$ from t corresponding to the attributes specified in the list.

Chapter Organization

- Intro & Terminologies
- RM formally defined (Set Theory basis)
- Components of Relational Model
- **Constraints on RM**
  - Basic Constraints
  - Key
  - Foreign Key

Constraints on Relational Model:

State of The DB: State of all Individual relation at a particular point in time.

DB represents a "Miniworld"

The rules and operation of this Miniworld, often imposes several constraints on the permissible state of the DB.

Constraints on Relational Model:

1. **Implicit Constraints**
   These are inherent to the data model. To be in RM, you have to adhere to them

2. **Explicit Constraints** [Schema based Constraints]
   Can be directly specified in the schema of the data model.

   DDL is used for this purpose.

3. **Application / Sematic Constraints** [Business Rules]
   Cant be expressed in the schema.

Constraints

Application Constraint

Is Total Participation an Application Constraint ?

Yes, Total participation is an application constraint.
RM doesn't have any provision to directly enforce Total participation constraints.

Implicit Constraints

Implicit Constraint	Description
No Duplicate Tuples	Each tuple must be unique in a relation.
Atomicity of Attributes (1NF)	Each attribute must store atomic values (no multi-valued or composite attributes).
Tuples are Unordered	The order of tuples in a relation does not matter.
Attributes are Unordered	The order of attributes in a relation does not matter.
Unique Attribute Names	No two attributes in the same relation can have the same name.

These are simply the rules of the Relational model

Explicit Constraints

1. Domain Constraint → Atomic, Should be within the specified range / format
2. Key Constraint
3. Entity Integrity Constraint
4. Referential Integrity

2. Key Constraint

By Definition: Relation is a **set** of tuples

No Duplicates

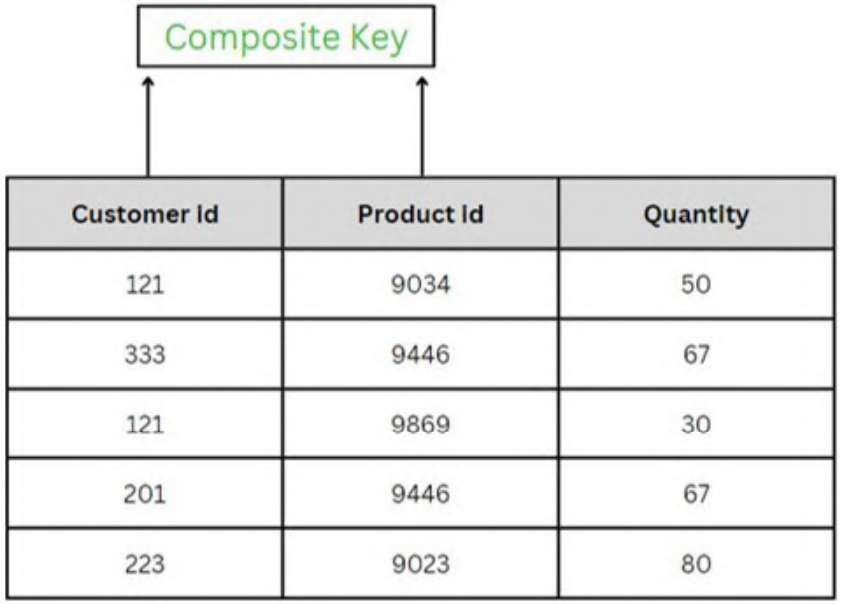
➔ No two rows can have the same combination of values for all its attributes.

NOTE: A "subset of Attributes" may also hold this property.

Super Key

$$t_1[\mathrm{SK}] \neq t_2[\mathrm{SK}]$$

If two or more attributes (columns) are involved in a key, then it is called as **Composite Key**



Customer Id	Product Id	Quantity
121	9034	50
333	9446	67
121	9869	30
201	9446	67
223	9023	80

Super keys are **in general** composite keys

Super Key

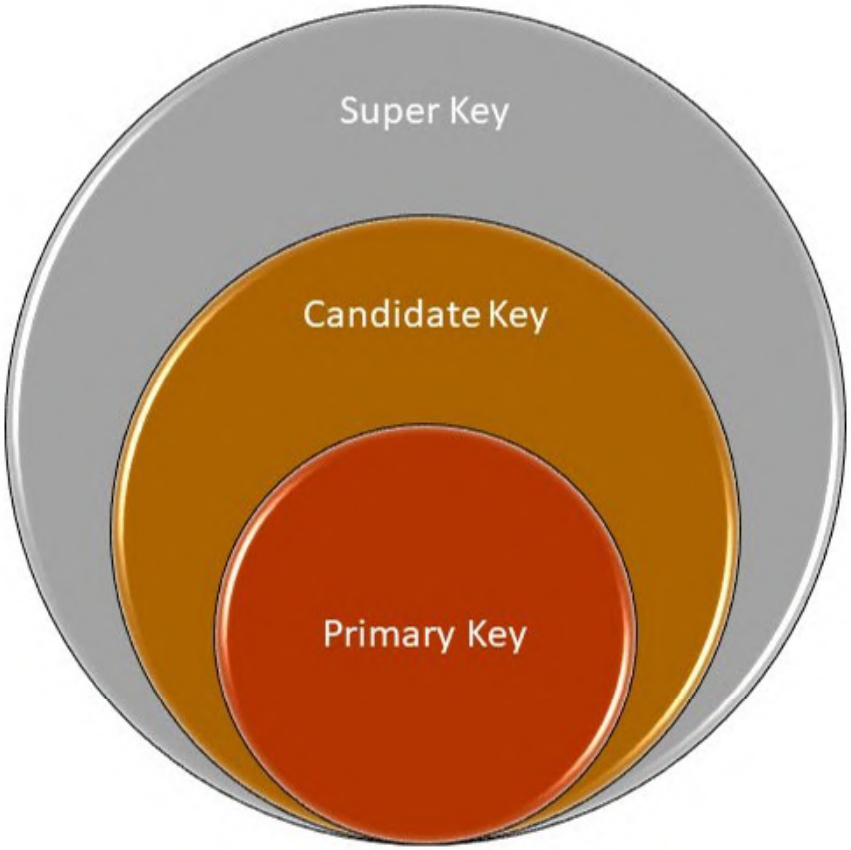
The entire schema (i.e. all attributes together) is always a superkey.

Superkey can have **redundant** attributes.

Go for Minimal Superkeys [**Candidate Keys**]

Remove any of its attributes and its no more a Valid key

Of all the candidate keys, one could be chosen as a **Primary Key** by the DBA.


Super Key
Candidate Key
Primary Key

KEY

- It is derived from the meaning of the attribute.
- It is time invariant.
- Key is a property of Relation Schema.

A key should hold on every valid relation state of the schema

NO Duplicate Tuples ⟷ **Key Constraint**

Though rows are guaranteed to be unique,
Key constraint is asking you to Explicitly declare a subset of Attributes as the Primary Key.

Though you know, rows are unique, its your job to choose and declare some attributes as the Primary Key.

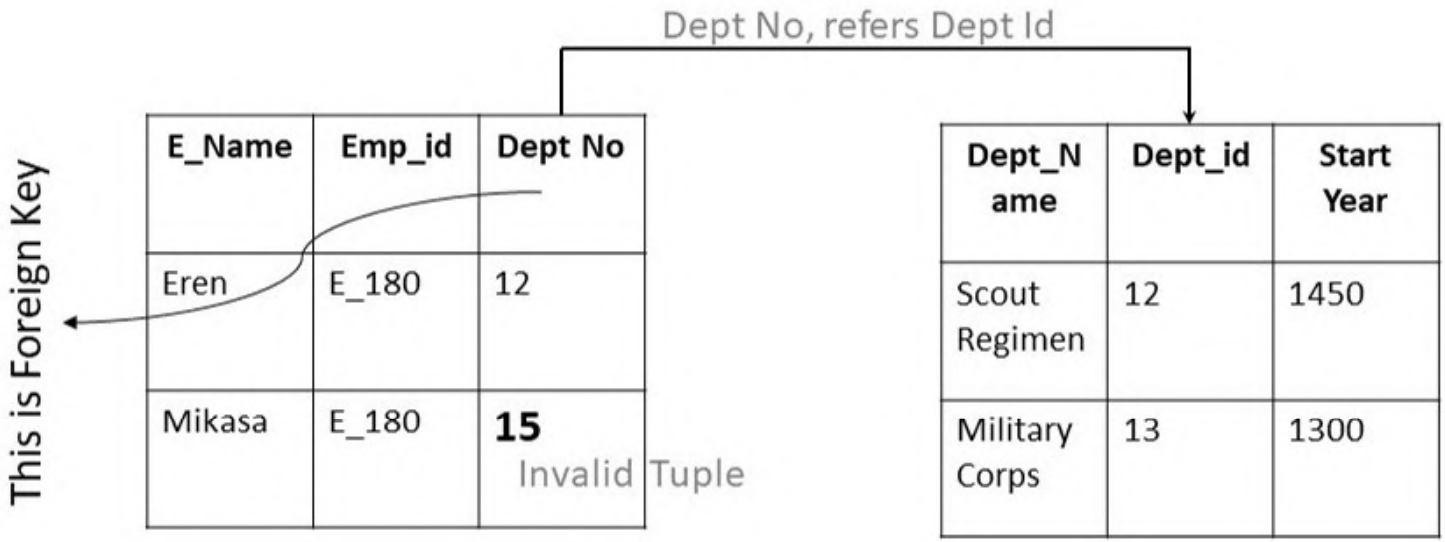
3. Entity Integrity Constraint

- No primary value can be NULL
- Also, no part of the primary value can be NULL

All the constraints we discussed till now are worried only about Single relation

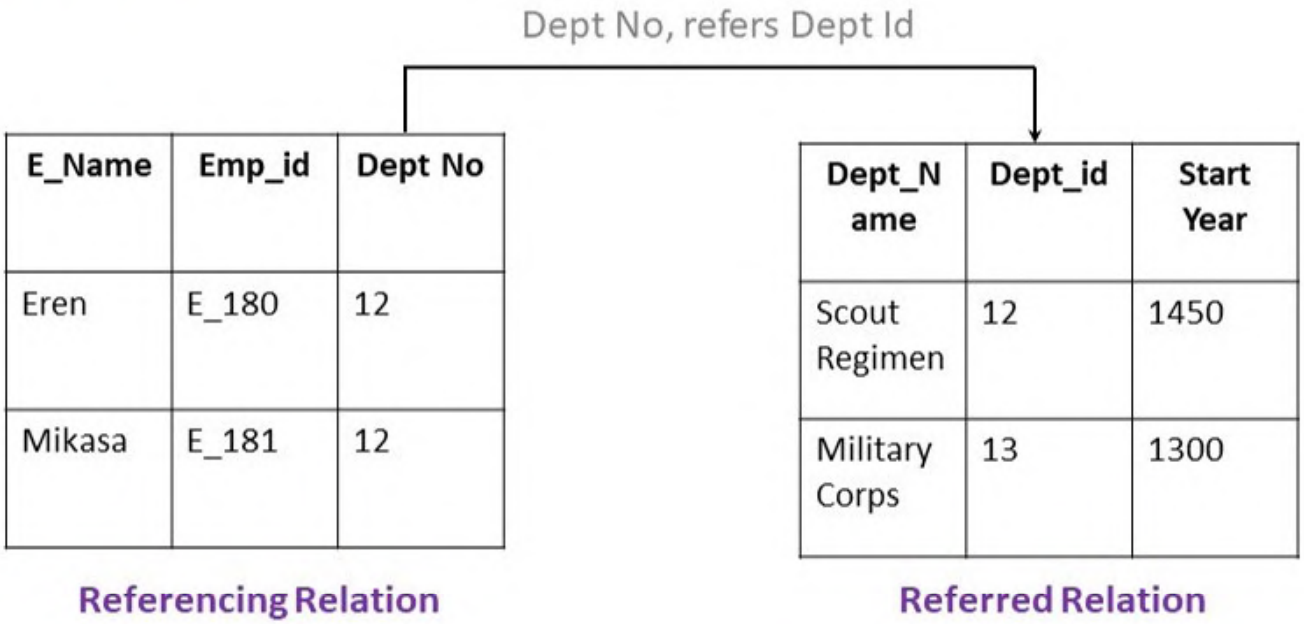
4. Referential Integrity Constraint

- Specified between two relations
- Helps to maintain consistency between tables



A tuple in one relation that refers to another relation must refer to an "existing tuple" in that other relation

Foreign Key



E_Name	Emp_id	Dept No
Eren	E_180	12
Mikasa	E_181	12

Referencing Relation

Dept_Name	Dept_id	Start Year
Scout Regimen	12	1450
Military Corps	13	1300

Referred Relation

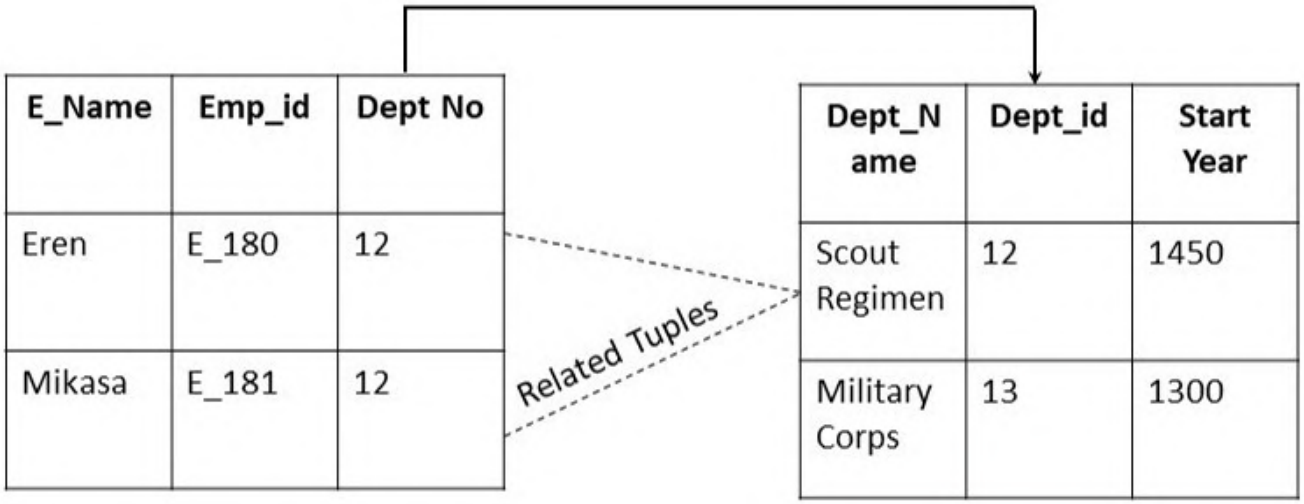
Note: Dept_Id is a Primary Key

A F.Key should always refer to an attribute, which is a P.Key in that referred table.

Domain (F.Key) == Domain(P.Key(R_2))

Foreign Key (Table 1) $\subset$ Primary Key (Table 2)

You cant refer to a Department that doesn't exist.
F.Key helps to eliminate many inconsistencies.



Tuple <Eren,E_180,12> and <Mikasa,E_181,12> **refers to** one “Scout_regimen” tuple.

F.Key helps to combine information in a meaningful way.

F.Key helps to answer queries involving two tables

In which year was the department where Eren works established?

Foreign Key in SQL

```
CREATE TABLE Department(
D_No INT PRIMARY KEY ,
D_Name VARCHAR (100)
);

CREATE TABLE Employee(
Emp_ID INT PRIMARY KEY,
Emp_Name VARCHAR (100),
Dept_ID INT,
FOREIGN KEY (Dept_ID) REFERENCES Department (D_No)
);
```

In this side, you declare that I'm pointing to an attribute Of the other table.

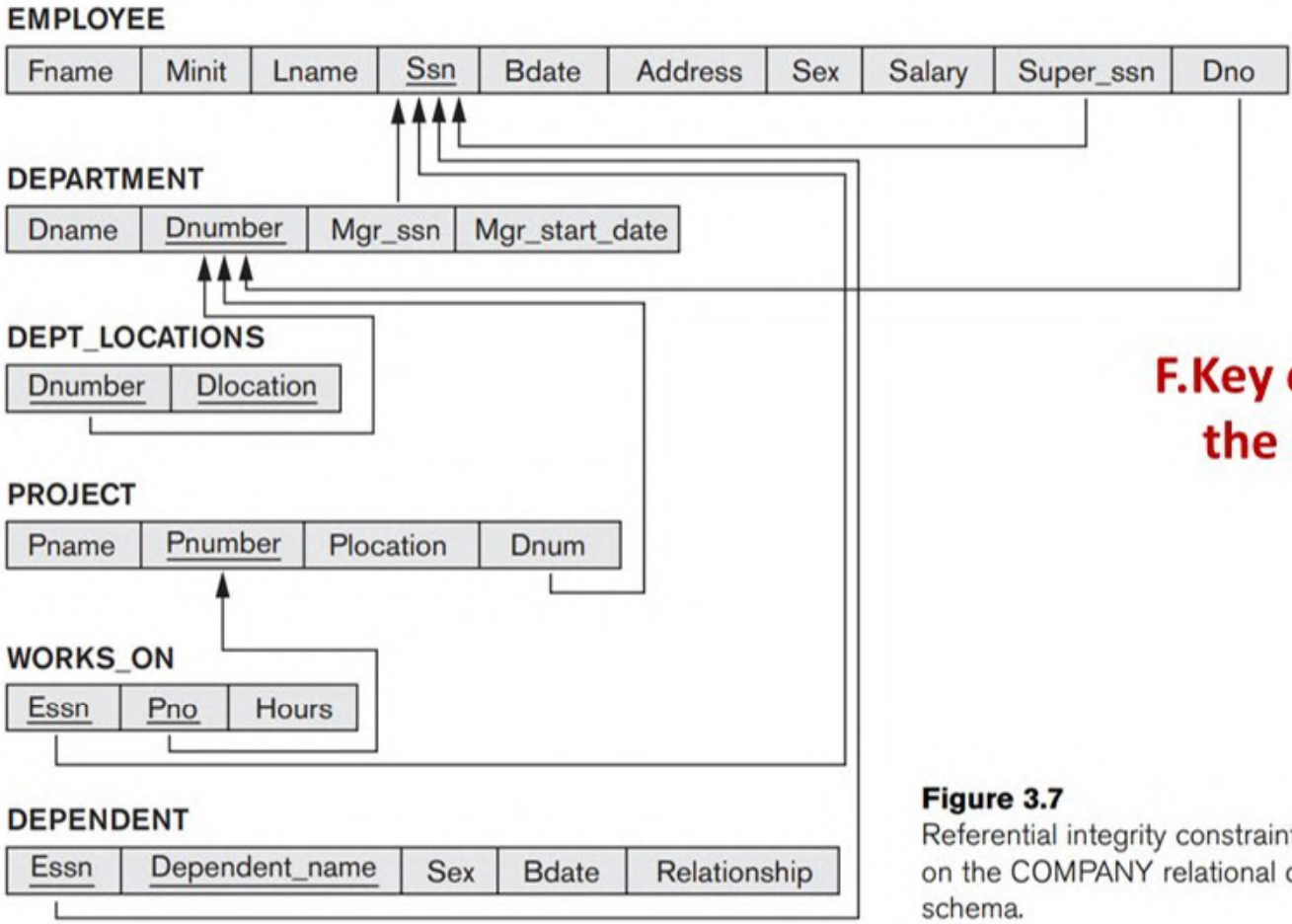
What is special about this Line ?
It is more like a pointer in C

To define referential integrity more formally, first we define the concept of a *foreign key*. The conditions for a foreign key, given below, specify a referential integrity constraint between the two relation schemas R_1 and R_2. A set of attributes FK in relation schema R_1 is a **foreign key** of R_1 that **references** relation R_2 if it satisfies the following rules:

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

In this definition, R_1 is called the **referencing relation** and R_2 is the **referenced relation**. If these two conditions hold, a **referential integrity constraint** from R_1 to R_2 is said to hold. In a database of many relations, there are usually many referential integrity constraints.

Chapter 3, Section 3.2.4, Page 73,Navathe



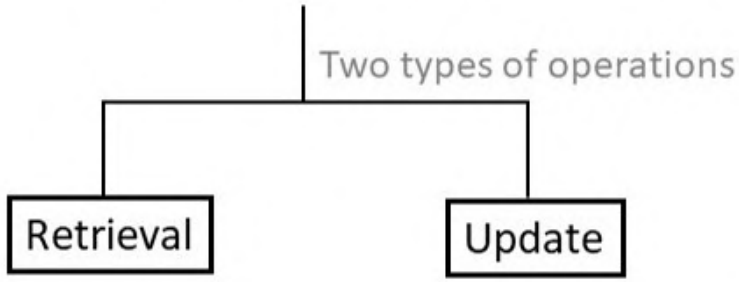
F.Key can be within the same table

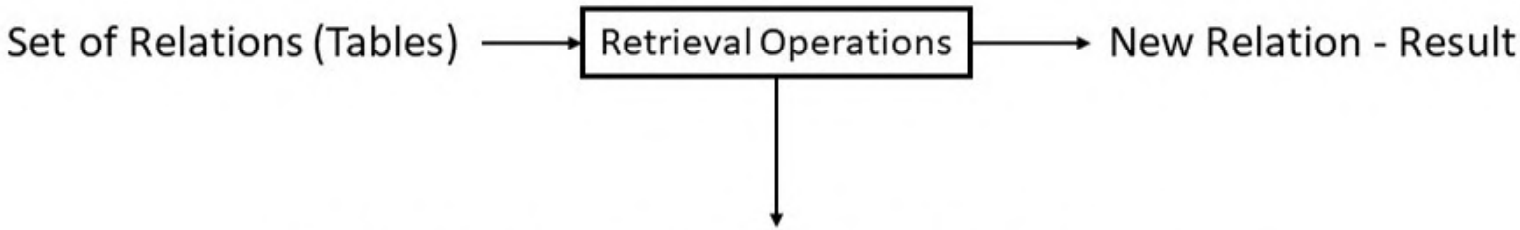
Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

Constraint Violation

Constraint Violation

There are operations which can violate constraints.





They do not affect any kind of Constraints

Update Operations

They can potentially violate Integrity Constraints (IC)

- Insert
- Delete
- Update (or Modify)

} Whenever they are performed IC's should not fail

Insert Operation

Insert operation provides a list of attribute-values for a new tuple t, which is to be inserted into R.

It can Violate all 4 Constraints

- Domain Constraint
- Key Constraint
- Entity Integrity Constraint
- Referential Integrity

Response / Countermeasure

Reject the insertion.

- Provide reason for rejection.
- Raise an prompt, to direct the user to modify their input.

If Referential Integrity is violated one option is to:

- Set it to NULL

Delete Operation

It can violate only Referential Integrity

Counter Measures

- Reject Deletion
- Cascade (Propagate deletion to referring table)
- Set Null (Problem if part of P.Key)

In DB development stage, when F.Key is specified,
the DBA also specifies which strategy to follow if RIC is violated.

Update Operation

No problem if it is a Non-Key attribute.
Problem only if it is a key attribute or F.Key

Counter Measures

Update == Delete + Insert