Database Management Systems(BCS403)

Module 2

Module – 2 Introduction to Relational Model

Contents

Relational Model:

- Relational Model Concepts
- Relational Model Constraints
- Relational database schemas
- Update operations and Transactions
- Dealing with constraint violations.

Relational Algebra:

- Unary and Binary relational operations
- Additional relational operations (aggregate, grouping, etc.)
- Examples of Queries in relational algebra.

Module – 2 Introduction to Relational Model

Contents

Mapping Conceptual Design into a Logical Design:

- Relational Database Design using ER-to-Relational mapping.

After learning this module, students should be able to

- Understand basic principles of the relational model of data.
- Define the modeling concepts and notation of the relational model.
- Define the update operations of the relational model, and understand how violations of integrity constraints are handled.

Module – 2 Introduction to Relational Model

Introduction

- In the mid-1960s and continuing through the 1970s and 1980s, early systems(large and expensive mainframe computers) were using database systems which were based on three main paradigms: hierarchical systems, network model–based systems, and inverted file systems.
- The relational data model was first introduced by **Ted Codd** of IBM Research in 1970.
- It became popular due to its **simplicity** and **mathematical foundation**.
- The model uses the concept of a **mathematical relation**(which looks somewhat like a table of values) as its basic building block, theoretical concepts of set theory and first-order predicate logic.

Module – 2 Introduction to Relational Model

Introduction

- The first commercial implementations of the relational model became available in the early 1980s, such as the **SQL/DS** system on the MVS(Multiple Virtual Storage) operating system by IBM and the **Oracle DBMS**.
- Since then, the model has been implemented in a large number of **commercial systems**, as well as a number of **open source systems**.
- Current popular commercial relational DBMSs (RDBMSs) include **DB2** (from IBM), **Oracle** (from Oracle), **Sybase** DBMS (now from SAP), and **SQLServer** and **Microsoft** Access (from Microsoft).
- In addition, several open source systems, such as MySQL and PostgreSQL, are available.

Module – 2 Introduction to Relational Model

Relational Model Concepts

- The **relational model** represents the database as a **collection of relations.**
- Informally, each **relation** resembles a **table of values** or, a **flat file of records**.
- It is called a flat file because each record has a simple linear or flat structure.
- **For example**, the university database of files(STUDENT, COURSE, SECTION, GRADE_REPORT AND PREREQUISITE) is similar to the basic relational model representation.
- However, there are **important differences between relations and files(table)** described below**;**

Module – 2 Introduction to Relational Model

Relational Model Concepts

Table or flat file

- When a relation is thought of as a **table of values**, each row in the table represents a collection of related data values.
- A **row represents** a fact that typically corresponds to a real-world entity or relationship.
- The **table name** and **column names** are used to help to interpret the meaning of the values in each row.
- **For example**, the first table (file) of university database is called STUDENT because each row represents facts about a particular **student** entity.
- The column names such as Name, Student_number, Class, and Major specify how to interpret the data values in each row.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Relation

In the formal relational model terminology,

- A row is called a **tuple**.
- A column header is called an **attribute**.
- The table is called a **relation**.
- So, the relation is a set of tuples.
- The **data type** describing the types of values that can appear in each column is represented by a **domain** of possible values.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Domains, Attributes, Tuples, and Relations

Domain

- A **domain** D is a set of atomic values.
- Atomic mean that each value in the domain is indivisible.
- A common method of specifying a domain is to specify a data type from which the data values are drawn.
- It is also useful to specify a name for the domain, to help in interpreting its values.

Some examples of domains follow:

- **Usa_phone_numbers**. The set of ten-digit phone numbers valid in the United States.

Module – 2 Introduction to Relational Model

Relational Model Concepts

- **Social_security_numbers.** The set of valid nine-digit Social Security numbers.
- **Names:** The set of character strings that represent names of persons.
- **Grade_point_averages**. Possible values of computed grade point averages; each must be a real (floating-point) number between 0 and 4.
- **Employee_ages.** Possible ages of employees in a company; each must be an integer value between 15 and 80.
- **Academic_department_names**. The set of academic department names in a university, such as Computer Science, Economics, and Physics.

Module – 2 Introduction to Relational Model

Relational Model Concepts

- A data type or format is also specified for each domain.
- **For example**, the data type for the domain **Usa_phone_numbers** can be declared as a character string of the form (ddd)ddd-dddd.
- The data type for **Employee_ages** is an integer number between 15 and 80.
- For **Academic_department_names**, the data type is the set of all character strings that represent valid department names.
- A **domain** is thus given a name, data type, and format.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Relation schema

- A **relation schema** R, denoted by $R(A_1, A_2, \ldots, A_n)$, is made up of a relation name R and a list of attributes, $A_1, A_2, \ldots, A_n$.
- Each attribute A_i is the name of a role played by some domain D in the relation schema R.
- D is called the domain of A_i and is denoted by $dom(A_i)$.
- A relation schema is used to describe a relation.
- R is called the name of this relation.
- The **degree** (or arity) of a relation is the **number of attributes** (denoted by n) of its relation schema.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Relation schema

$$R(A_1, A_2, A_3, A_4, A_5, A_6, A_7)$$

STUDENT(Name, Ssn, Home_phone, Address, Office_phone, Age, Gpa)

Degree of a relation STUDENT = 7

Using the data type of each attribute, the definition is sometimes written as:

STUDENT(Name: string, Ssn: string, Home_phone: string, Address: string, Office_phone: string, Age: integer, Gpa: real)

Module – 2 Introduction to Relational Model

Relational Model Concepts

Relation schema

Also we can **specify the domains** for some of the attributes of the STUDENT relation:

dom(Name) = Names; dom(Ssn) = Social_security_numbers; dom(HomePhone) = USA_phone_numbers , dom(Office_phone) = USA_phone_numbers, and dom(Gpa) = Grade_point_averages.

Relation state

- A relation (or relation state) **r** of the relation schema R(A_1, A_2, … , A_n), also denoted by **r(R)**, is a set of **m**-tuples r = {t_1, t_2, … , t_m}.
- Each **m**-tuple **t** is an ordered list of **n** values t =<v_1, v_2,….v_n>, where each value v_i , $1 \leq i \leq n$, is an element of dom (Ai) or is a special NULL value.

Module – 2 Introduction to Relational Model

Relational Model Concepts

- The i^{th} value in tuple t, which corresponds to the attribute A_i , is referred to as $t[A_i]$ or $t.A_i$
- The relation schema $\mathbf{R}(A_1, A_2, \ldots, A_n)$ is termed as **intension.**
- The relation state r(**R**) is termed as **extension**.

The definition of a relation can be **restated more formally** using **set theory** concepts as follows.

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

A relation (or relation state) **r**(**R**) is a mathematical relation of degree **n** on the domains $dom(A_1)$, $dom(A_2)$, … , $dom(A_n)$, which is a subset of the Cartesian product of the domains that define **R**.

Module – 2 Introduction to Relational Model

Relational Model Concepts

- 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

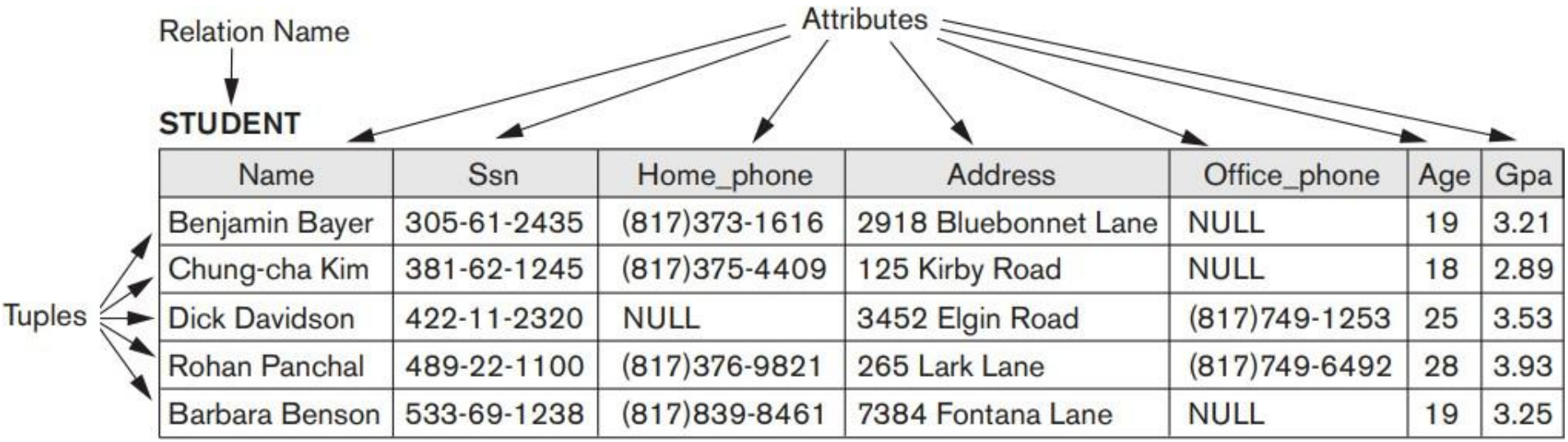
$$|dom(A_1)| \times |dom(A_2)| \times \ldots \times |dom(A_n)|$$

- 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**).

Module – 2 Introduction to Relational Model

Relational Model Concepts

Relation schema



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

Module – 2 Introduction to Relational Model

Relational Model Concepts

Characteristics of Relations

Ordering of Tuples in a Relation

- A relation is not sensitive to the ordering of tuples. However, in a file, records are physically stored on disk (or in memory), so there always is an order among the records.
- Tuple ordering is not part of a relation definition because a relation attempts to represent facts at a logical or abstract level.

Ordering of Values within a Tuple

- At a more abstract level, the **order of attributes** and their **values** is not that important as long as the correspondence between attributes and values is maintained.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Characteristics of Relations

- Making the ordering of values in a tuple **unnecessary**. In this definition, a relation schema **R** = {A_1, A_2, … , A_n} is a set of attributes (instead of an ordered list of attributes).
- A relation state r(**R**) is a finite set of mappings r = {t_1, t_2, … , t_m}, where each tuple t_i is a mapping from **R** to **D**, and **D** is the union of the attribute domains; that is, **D** = dom(A_1) U dom(A_2) U … U dom(A_n).

t = < (Name, Dick Davidson),(Ssn, 422-11-2320),(Home_phone, NULL),(Address, 3452 Elgin Road),
(Office_phone, (817)749-1253),(Age, 25),(Gpa, 3.53)>

t = < (Address, 3452 Elgin Road),(Name, Dick Davidson),(Ssn, 422-11-2320),(Age, 25),
(Office_phone, (817)749-1253),(Gpa, 3.53),(Home_phone, NULL)>

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

Module – 2 Introduction to Relational Model

Relational Model Concepts

Characteristics of Relations

Values and NULLs in the Tuples

- Each value in a tuple is an atomic value; that is, it is not divisible into components within the framework of the basic relational model.
- Hence, composite and multivalued attributes are not allowed. This model is sometimes called the flat relational model.
- A special value, called **NULL**, is used when the values of attributes that may be unknown or may not apply to a tuple.
- We can have **several meanings** for **NULL** values, such as **value unknown**, **value exists but is not available**, or **attribute does not apply** to this tuple (also known as value undefined).

Module – 2 Introduction to Relational Model

Relational Model Concepts

Characteristics of Relations

Interpretation (Meaning) of a Relation

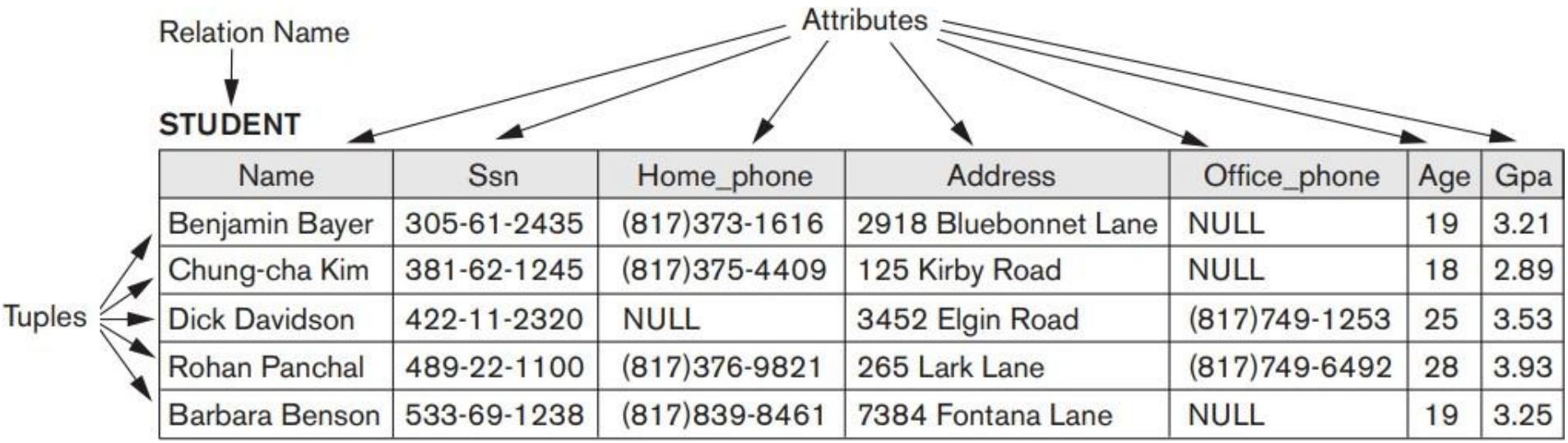
- The relation schema can be interpreted as a **declaration** or a type of **assertion**.
- Consider a student entity which has a Name, Ssn, Home_phone, Address, Office_phone, Age, and Gpa.
- Each tuple in the relation can then be **interpreted as** a fact or a particular instance of the assertion.
- **For example**, the first tuple in figure shown below asserts the fact that there is a STUDENT whose Name is Benjamin Bayer, Ssn is 305-61-2435, Age is 19, and Gpa is 3.21.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Characteristics of Relations

Interpretation (Meaning) of a Relation



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

Note: some relations may represent facts about entities, whereas other relations may represent facts about relationships.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Characteristics of Relations

Interpretation (Meaning) of a Relation

- **For example,** a relation schema MAJORS (Student_ssn, Department_code) asserts that students major in academic disciplines.
- A tuple in this relation **relates** a student to his or her major discipline.
- Hence, the **relational model represents facts about both entities and relationships** uniformly as relations.

Majors

Student_ssn	Department_code
305-61-2435	CS
381-62-1245	CS
422-11-2320	ECE

Module – 2 Introduction to Relational Model

Relational Model Concepts

Relational Model Notation

- A relation schema **R** of degree **n** is denoted by $R(A_1, A_2, \ldots, A_n)$.
- 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.
- An n-tuple t in a relation r(R) is denoted by t = , where v_i is the value corresponding to attribute Ai.
- 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.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Relational Model Constraints and Relational Database Schemas

- In a relational database, there will typically be many relations, and the tuples in those relations are usually related in various ways.
- The state of the whole database will correspond to the states of all its relations at a particular point in time.
- There are generally many restrictions or constraints on the actual values in a database state.
- These constraints are derived from the **rules in the mini-world** that the database represents.
- Constraints on databases can generally be divided into **three main categories:**

Module – 2 Introduction to Relational Model

Relational Model Concepts

Relational Model Constraints and Relational Database Schemas

1. Constraints that are **inherent in the data model**. These are called as **inherent model-based constraints** or **implicit constraints**.
2. Constraints that can be **directly expressed in the schemas** of the data model, typically by **specifying them in the DDL. These are** called as **schema-based constraints** or **explicit constraints**.
3. Constraints that **cannot be directly expressed** in the schemas of the data model, and hence must be expressed and **enforced by the application programs**. These are called as **application-base**d or **semantic constraints** or **business rules**.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Inherent model-based constraints or implicit constraints

- The characteristics of relations are the inherent constraints of the relational model.
- **For example**, the constraint that a relation cannot have duplicate tuples is an inherent constraint.

Schema-based constraints or explicit constraints

The schema-based constraints include

1. domain constraints,
2. key constraints,
3. constraints on NULLs,
4. Entity integrity constraints, and Referential integrity constraints.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Domain Constraints

- Domain constraints specify that within each tuple, the value of each attribute A must be an atomic value from the domain **dom**(A).
- The data types associated with domains typically include standard numeric data types for integers and real numbers (float and double-precision float).
- Characters, Booleans, fixed-length strings.
- Variable-length strings are also available, as are date, time, timestamp, and other special data types.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Key Constraints and Constraints on NULL Values

- A relation is defined as a set of tuples, and we know that no two tuples can have the same combination of values for all their attributes.

Super key(SK)

- There are other **subsets of attributes** of a relation schema **R** with the property **that no two tuples** in any relation state **r** of **R** should have the same combination of values for these attributes.
- This subset of attributes is called as **super key(SK)**.
- For any two distinct tuples t_1 and t_2 in a relation state r of **R**, we have the constraint that:

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

Module – 2 Introduction to Relational Model

Relational Model Concepts

Key Constraints and Constraints on NULL Values

- A super key **SK** specifies a **uniqueness constraint** that no two distinct tuples in any state r of **R** can have the same value for SK.
- **For example** the super key(SK) = {book_id, branch_id}

r(book_copies) =

book_id	branch_id	no_of_copies
124	SDM01	25
124	SDM02	40
128	SDM01	5
128	SDM02	32

Module – 2 Introduction to Relational Model

Relational Model Concepts

Key Constraints and Constraints on NULL Values

Key: A key **k** of a relation schema **R** is a super-key of **R** with the additional property that removing any attribute **A** from **K(**License_number,Engine_serial_number**)** leaves a set of attributes **K′** that is not a super-key of **R** any more.

key satisfies two properties:

1. Two distinct tuples in any state of the relation **cannot have identical values** for (all) the attributes in the key. This uniqueness property also applies to a super-key.
2. It is a **minimal super-key**, that is, a super-key from which we cannot remove any attributes and still have the uniqueness constraint hold. This minimality property is required for a key but is optional for a super-key.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Key Constraints and Constraints on NULL Values

- Hence, a key is a super-key but **not vice versa**.
- A **super-key may be a key** (if it is minimal) or **may not be a key** (if it is not minimal).

Example:

- The attribute set {Ssn} is a key of relation STUDENT because no two student tuples can have the same value for Ssn.
- However, the super-key {Ssn, Name, Age} is not a key of STUDENT because removing Name or Age or both from the set still leaves us with a super-key.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Key Constraints and Constraints on NULL Values

- In general, a relation schema may have **more than one key**. In this case, each of the keys is **called a candidate key**.
- It is common to designate **one of the candidate keys as the primary key of the relation**. This is the candidate key whose values are used to identify tuples in the relation.

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

CAR

License_number	Engine_serial_number	Make	Model	Year
Texas ABC-739	A69352	Ford	Mustang	02
Florida TVP-347	B43696	Oldsmobile	Cutlass	05
New York MPO-22	X83554	Oldsmobile	Delta	01
California 432-TFY	C43742	Mercedes	190-D	99
California RSK-629	Y82935	Toyota	Camry	04
Texas RSK-629	U028365	Jaguar	XJS	04

Module – 2 Introduction to Relational Model

Relational Model Concepts

Relational Databases and Relational Database Schemas

- A relational database usually contains many relations, with tuples in relations that are related in various ways.
- A relational database schema **S** is a set of relation schemas **S** = {R_1, R_2, … , R_m} and a set of integrity constraints **IC**.
- A relational database state **DB** of **S** is a set of relation states DB = {r_1, r_2, … , r_m} such that each r_i is a state of R_i and such that the r_i relation states satisfy the integrity constraints specified in IC.

For example COMPANY = {EMPLOYEE, DEPARTMENT, DEPT_LOCATIONS, PROJECT, WORKS_ON, DEPENDENT}.

Module – 2 Introduction to Relational Model

Relational Model Concepts

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
Schema diagram for the COMPANY relational database schema.

Module – 2 Introduction to Relational Model

Relational Model Concepts

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: One possible database state for the COMPANY relational database schema.

Module – 2 Introduction to Relational Model

Relational Model Concepts

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

Module – 2 Introduction to Relational Model

Relational Model Concepts

- A database state that **does not obey all the integrity constraints** is called not valid, and a state that **satisfies all the constraints** in the defined set of integrity constraints IC is called a valid state.

Note:

- Attributes that represent the same real-world concept may or may not have identical names in different relations.
- **For example** the Dnumber attribute in both **DEPARTMENT** and **DEPT_LOCATIONS** stands for the same real-world concept the number given to a department.
- **Alternatively**, attributes that represent different concepts may have the same name in different relations.
- **For example**, the attribute name Name can be used for both Pname of PROJECT and Dname of DEPARTMENT;

Module – 2 Introduction to Relational Model

Relational Model Concepts

Entity Integrity, Referential Integrity, and Foreign Keys

- The **entity integrity constraint** states that no primary key value can be NULL, because the primary key value is used to identify individual tuples in a relation.
- Having NULL values for the primary key implies that we cannot identify some tuples.
- For example, if two or more tuples had NULL for their primary keys, we may not be able to distinguish them if we try to reference them from other relations.
- **Key constraints** and **entity integrity** constraints are specified on **individual relations**.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Entity Integrity, Referential Integrity, and Foreign Keys

- The **referential integrity constraint** is specified between two relations and is used to **maintain the consistency** among tuples in the two relations.
- **Informally**, the referential integrity constraint states that a tuple in one relation that refers to another relation must refer to an existing tuple in that relation.
- **Formally**, **t**he conditions for a **foreign key**, given below, specify a **referential integrity constraint** between the two relation schemas R1 and R2.
- A set of attributes **FK** in relation schema **R2** is a foreign key of **R2** that references relation **R1** if it satisfies the following rules:

Module – 2 Introduction to Relational Model

Relational Model Concepts

Entity Integrity, Referential Integrity, and Foreign Keys

1. The attributes in **FK** have the same domain(s) as the primary key attributes **PK** of R1; the attributes FK are said to reference or refer to the relation R1.

2. A value of **FK** in a tuple **t1** of the current state r(**R2**) either occurs as a value of **PK** for some tuple **t2** in the current state r(**R1**) or is **NULL**. That is t1[FK] = t2[PK], and we say that the tuple t1 references or refers to the tuple t2.

In this definition, **R2** is called the referencing relation and **R1** is the referenced relation. If the above two conditions satisfied, a referential integrity constraint from **R2** to **R1** is said to hold.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Entity Integrity, Referential Integrity, and Foreign Keys

r(R1) = r(publisher) **(Primary key)**

name	address	phone
McGraw	Delhi	9845098450
Pearson	Mangaluru	9741970005

(Foreign key)

r(R2) = r(book)

book_id	title	PUBLISHER_name	pub_year
124	OS	Pearson	2017
125	DBMS	McGraw	2013
128	Algorithms	Pearson	2018

Module – 2 Introduction to Relational Model

Relational Model Concepts

Entity Integrity, Referential Integrity, and Foreign Keys

Note:

- A foreign key **can refer to its own relation**.
- **For example**, the attribute Super_ssn in EMPLOYEE refers to the supervisor of an employee; this is another employee, represented by a tuple in the EMPLOYEE relation. Hence, Super_ssn is a foreign key that references the EMPLOYEE relation itself.
- Referential integrity constraints are diagrammatically displayed by drawing a **directed arc** from each foreign key to the relation it references.
- This type of constraints can be called as **state constraints** because they define the constraints that a valid state of the database must satisfy.

Module – 2 Introduction to Relational Model

Relational Model Concepts

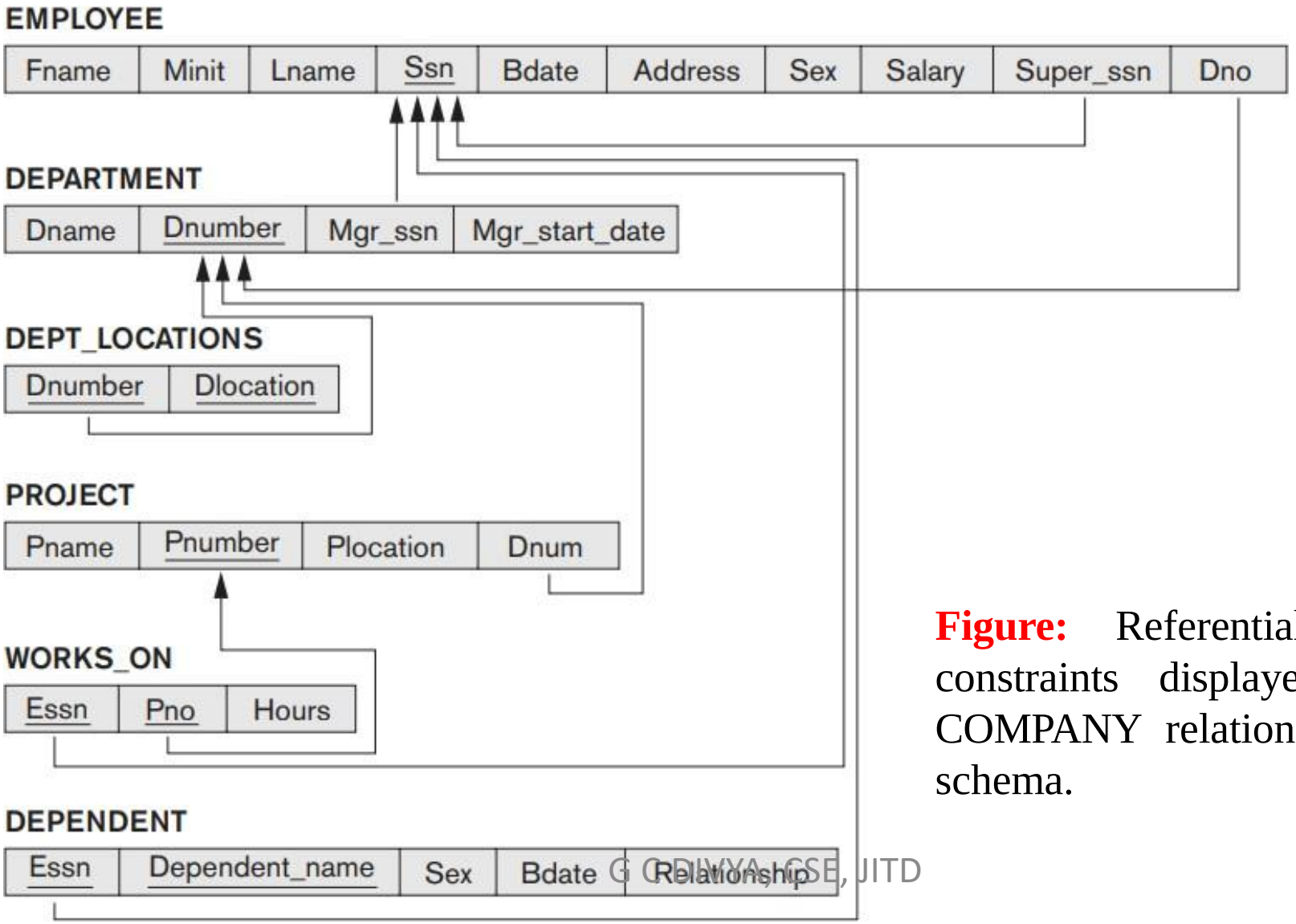


Figure: Referential integrity constraints displayed on the COMPANY relational database schema.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Application-based or **semantic constraints** or **business rules**.

- These are not part of the DDL and have to be specified and enforced in a different way.
- **Example:** the salary of an employee should not exceed the salary of the employee‘s supervisor and the maximum number of hours an employee can work on all projects per week is 56.
- Such constraints can be specified and enforced within the application programs that update the database, or by using a general-purpose constraint specification language.
- This type of constraints can be called as **transition constraints** because which deal with state changes in the database.

Module – 2 Introduction to Relational Model

Relational Model Concepts

r(publisher)

name	address	phone
McGraw	Delhi	9845098450
Pearson	Mangaluru	9741970005

r(book)

book_id	title	PUBLISHER_name	pub_year
124	OS	Pearson	2017
125	DBMS	McGraw	2013
128	Algorithms	Pearson	2018

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

The operations of the relational model can be categorized into the following

1. Retrievals
2. Updates.

- The relational algebra operations, which can be used to specify retrievals.
- A relational algebra expression forms a new relation after applying a number of algebraic operators to an existing set of relations.
- The main use of relational algebra expression is for querying a database to retrieve information.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

Database modification or update operations

There are **three basic operations** that can change the states of relations in the database:

1. Insert
2. Delete
3. Update (or Modify)

Insert is used to insert one or more new tuples in a relation.

Delete is used to delete tuples.

Update (or Modify) is used to change the values of some attributes in existing tuples.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

Note: Whenever these operations are applied, the **integrity constraints** specified on the relational database schema should not be violated.

The Insert Operation

- The Insert operation provides a list of attribute values for a new tuple **t** that is to be inserted into a relation R.
- Insert can violate any of the **four types** of constraints.
- **Domain constraints** can be violated if an attribute value is given that does not appear in the corresponding domain or is not of the appropriate data type.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

```
MySQL  localhost:33060+ ssl  testdb  SQL > insert into publisher values("Sapna
","Bengaluru","abcd");
ERROR: 1366: Incorrect integer value: 'abcd' for column 'phone' at row 1
```

Key constraints can be violated if a key value in the new tuple **t** **already exists** in another tuple in the relation r(R).

```
MySQL  localhost:33060+ ssl  testdb  SQL > insert into publisher values("pears
on","Bengaluru","9741970054");
ERROR: 1062: Duplicate entry 'pearson' for key 'publisher.PRIMARY'
```

Entity integrity can be violated if any part of the primary key of the new tuple **t** is NULL.

```
MySQL  localhost:33060+ ssl  testdb  SQL > insert into publisher values(NULL,"
Bengaluru",2233455);
ERROR: 1048: Column 'name' cannot be null
```

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

Referential integrity can be violated if the value of any foreign key in **t** refers to a tuple that does not exist in the referenced relation.

```
MySQL  localhost:33060+ ssl  testdb  SQL > insert into book values(127,"Hadoop
","Sapna",2014);
ERROR: 1452: Cannot add or update a child row: a foreign key constraint fails (
`testdb`.`book`, CONSTRAINT `book_ibfk_1` FOREIGN KEY (`PUBLISHER_name`) REFERE
NCES `publisher` (`name`) ON DELETE CASCADE)
```

- If an insertion violates one or more constraints, the default option is to reject the insertion.
- It would be useful **if the DBMS could provide a reason** to the user as to why the insertion was rejected.
- **Another option** is to attempt to correct the reason for rejecting the insertion, but this is **typically not used for violations caused by Insert**; rather, it is used more often in correcting violations for Delete and Update.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

The Delete Operation

- The Delete operation can violate only **referential integrity**.
- This occurs if the tuple being deleted is referenced by foreign keys from other tuples in the database.
- To specify deletion, a condition on the attributes of the relation selects the tuple (or tuples) to be deleted.

```
MySQL  localhost:33060+ ssl  testdb  SQL > delete from publisher1 where name =
 "Sapna";
ERROR: 1451: Cannot delete or update a parent row: a foreign key constraint fai
ls (`testdb`.`book1`, CONSTRAINT `book1_ibfk_1` FOREIGN KEY (`publisher_name`)
```

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

- Several options are available if a deletion operation causes a violation.
- **The first option** is to reject the deletion.
- The **second option**, called **cascade(on delete cascade)**, is to attempt to cascade (or propagate) the deletion by deleting tuples that reference the tuple that is being deleted.
- A **third option**, called set null or set default, is to modify the referencing attribute values that cause the violation; each such value is either set to NULL or changed to reference another default valid tuple.
- **Notice that** if a referencing attribute that causes a violation is part of the primary key, it cannot be set to NULL; otherwise, it would violate entity integrity.

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

The Update Operation

- The Update (or Modify) operation is used to change the values of one or more attributes in a tuple (or tuples) of some relation **R**.
- It is necessary to specify a condition on the attributes of the relation to select the tuple (or tuples) to be modified.

```
MySQL  localhost:33060+ ssl  testdb  SQL > update book set publisher_name = "S
apna" where book_id = 124;
ERROR: 1452: Cannot add or update a child row: a foreign key constraint fails (
`testdb`.`book`, CONSTRAINT `book_ibfk_1` FOREIGN KEY (`PUBLISHER_name`) REFERE
NCES `publisher` (`name`) ON DELETE CASCADE)
```

```
MySQL  localhost:33060+ ssl  testdb  SQL > update book set publisher_name = "p
earson" where book_id = 125;
Query OK, 1 row affected (0.0041 sec)

Rows matched: 1  Changed: 1  Warnings: 0
```

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

```
MySQL  localhost:33060+ ssl  testdb  SQL > update book set book_id = 125 where
 book_id = 124;
ERROR: 1451: Cannot delete or update a parent row: a foreign key constraint fai
ls (`testdb`.`book_authors`, CONSTRAINT `book_authors_ibfk_1` FOREIGN KEY (`boo
k_id`) REFERENCES `book` (`book_id`) ON DELETE CASCADE)
```

- Updating an attribute that is neither part of a primary key nor part of a foreign key usually causes no problems; the DBMS need only check to confirm that the new value is of the correct data type and domain.

```
MySQL  localhost:33060+ ssl  testdb  SQL > update book set pub_year = 2015 whe
re book_id = 124;
Query OK, 1 row affected (0.0038 sec)

Rows matched: 1  Changed: 1  Warnings: 0
```

Module – 2 Introduction to Relational Model

Relational Model Concepts

Update Operations and Dealing with Constraint Violations

- Modifying a primary key value is similar to deleting one tuple and inserting another in its place because we use the primary key to identify tuples.
- When a referential integrity constraint is specified in the DDL, the **DBMS will allow the user** to choose separate options to deal with a violation caused by **Delete** and a violation caused by **Update**.

Module – 2 Introduction to Relational Model

Relational Model Concepts

The Transaction Concept

- A database **application program** running against a relational database typically executes one or more transactions.
- A **transaction** is an executing program that includes some database operations, such as reading from the database, or applying insertions, deletions, or updates to the database.
- At the end of the transaction, it must leave the database in a valid or consistent state that satisfies all the constraints specified on the database schema.
- A single transaction may involve any number of **retrieval** operations and any number of **update** operations.

Module – 2 Introduction to Relational Model

Relational Model Concepts

The Transaction Concept

- **For example**, a transaction to apply a bank withdrawal will typically read the user account record, **check** if there is a sufficient balance, and then update the record by the withdrawal amount.
- A large number of commercial applications running against relational databases in **online transaction processing (OLTP)** systems are executing transactions at rates that reach several hundred per second.

The Relational Algebra and Relational Calculus

Module – 2 Introduction to Relational Algebra

Introduction

- We know that, a data model defines the database‘s structure and constraints, also must include a set of operations to manipulate the database.
- The **basic set of operations** for the formal relational model is the **relational algebra**.
- These operations enable a user to specify basic retrieval requests as **relational algebra expressions**.
- The result of a retrieval query is a **new relation**. The algebra operations thus produce new relations, which can be **further manipulated using operations** of the same algebra.
- A sequence of relational algebra operations forms a **relational algebra expression.**

Module – 2 Introduction to Relational Algebra

Introduction

Why relational algebra is very important?

- **First,** it provides a formal foundation for relational model operations.
- **Second,** and more important, it is used as a basis for implementing and optimizing queries in the query processing and optimization modules that are integral parts of relational database management systems (RDBMSs).
- **Third,** some of its concepts are incorporated into the SQL standard query language for RDBMSs.

The **relational calculus** provides a higher-level declarative language for specifying relational queries.

Note: Most commercial RDBMSs in use today do not provide user interfaces for relational algebra queries.

Module – 2 Introduction to Relational Algebra

Introduction

- The relational algebra is often considered to be **an integral part** of the **relational data model**.

Its operations can be divided into two groups.

- **Group-1** includes **set operations from mathematical set theory**; these are applicable because each relation is defined to be a set of tuples in the formal relational model.
- Set operations include UNION, INTERSECTION, SET DIFFERENCE, and CARTESIAN PRODUCT (also known as CROSS PRODUCT).
- **Group-2** consists of operations developed specifically for relational databases these include SELECT, PROJECT, and JOIN.

Module – 2 Introduction to Relational Algebra

Unary Relational operations

- The operations that **operate on single relations** are called as unary operations.

  1. SELECT
  2. PROJECT

SELECT

- The **SELECT** operation is used to choose a subset of the tuples from a relation that satisfies a selection condition.
- We can consider the **SELECT** operation to be a filter that keeps only those tuples that satisfy a qualifying condition.

Module – 2 Introduction to Relational Algebra

Unary Relational operations

- The **SELECT** operation can also be visualized as a horizontal partition of the relation into **two sets of tuples.**
- **Set – 1**: those tuples that satisfy the condition and are selected.
- **Set – 2**: those tuples that do not satisfy the condition and are filtered out.

General form of unary selection operation

$$\sigma_{<\text{selection condition}>}(R)$$

- Where the symbol **σ (sigma)** is used to denote the **SELECT** operator.
- The selection condition is a **Boolean expression** (condition) specified on the attributes of relation R.

Module – 2 Introduction to Relational Algebra

Unary Relational operations

- **R** is generally a relational algebra expression whose result is a relation or just the name of a **database relation**.
- The relation resulting from the **SELECT** operation has the same attributes as **R**.
- **For example**, to select the EMPLOYEE tuples whose **department is 4**, or those whose **salary is greater than $30,000**.

$$\sigma_{Dno=4}(EMPLOYEE)$$

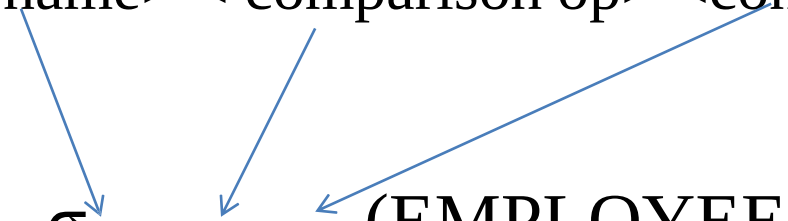
$$\sigma_{Salary>30000}(EMPLOYEE)$$

Module – 2 Introduction to Relational Algebra

Unary Relational operations

- The Boolean expression specified in<**select condition** >is made up of a number of clauses of the form;

<attribute name> < comparison op> <constant value>



$\sigma_{Salary>30000}(EMPLOYEE)$

Where **<attribute name>** is the name of an attribute of **R**,

< comparison op> is normally one of the operators {=, <, <=, >, >=, ≠},

<**constant value**> is a constant value from the **attribute domain**.

Module – 2 Introduction to Relational Algebra

Unary Relational operations

- Clauses can be connected by the standard Boolean operators and, or, and not to form a general selection condition.
- For example, to select the tuples for all employees who either work in department 4 and make over $25,000 per year, or work in department 5 and make over $30,000, the following SELECT operation can be used:

$$\sigma_{(Dno=4\ \textbf{AND}\ Salary>25000)\ \textbf{OR}\ (Dno=5\ \textbf{AND}\ Salary>30000)}(EMPLOYEE)$$

Equivalent SQL query

```
SELECT * FROM EMPLOYEE WHERE Dno=4 AND Salary>25000;
```

Output relation

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
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

Module – 2 Introduction to Relational Algebra

Unary Relational operations

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

Module – 2 Introduction to Relational Algebra

Unary Relational operations

- If the domain of an attribute is a set of unordered values, then only the comparison operators in the set {=, ≠} can be used.
- An example of an unordered domain is the domain Color = { ‗red‘, ‗blue‘, ‗green‘, ‗white‘, ‗yellow‘, …}

$$\sigma_{color = red}(CAR)$$

SELECT operation works as follows:

- The < **selection condition**> is applied independently to each individual tuple t in **R**.
- This is done by substituting each occurrence of an attribute $\mathbf{A_i}$ in the selection condition with its value in the tuple t[$\mathbf{A_i}$].
- If the condition evaluates to TRUE, then tuple t is selected.

Module – 2 Introduction to Relational Algebra

Unary Relational operations

- All the selected tuples appear in the result of the **SELECT** operation.

The Boolean conditions AND, OR, and NOT are interpreted as follows:

- (cond1 **AND** cond2) is TRUE if both (cond1) and (cond2) are TRUE; otherwise, it is FALSE.
- (cond1 **OR** cond2) is TRUE if either (cond1) or (cond2) or both are TRUE; otherwise, it is FALSE.
- (**NOT** cond) is TRUE if cond is FALSE; otherwise, it is FALSE.
- The **degree** of the relation resulting from a **SELECT** operation is its **number of attributes** that is the same as the degree of **R**.

Module – 2 Introduction to Relational Algebra

Unary Relational operations

- The number of tuples in the resulting relation is always less than or equal to the number of tuples in R.

i.e. $|\sigma c\ (R)| \leq |R|$ for any condition C.

The **SELECT** operation is **commutative**; that is

$$\sigma_{<condition1>}(\sigma_{<condition2>}(R)) = \sigma_{<condition2>}(\sigma_{<condition1>}(R))$$

Hence, a sequence of SELECTs can be applied in any order.

In addition, we can always combine a cascade (or sequence) of SELECT operations into a single SELECT operation with a conjunctive (AND) condition.

$$\sigma_{<condition1>}(\sigma_{<condition1>}(\ldots(\sigma_{<condition1>}(R))\ldots)) = \sigma_{<condition1>\ AND\ <condition1>\ AND \cdots AND\ <condition1>}(R)$$

Module – 2 Introduction to Relational Algebra

Unary Relational operations

The PROJECT Operation

- The **SELECT** operation chooses some of the **rows** from the table while discarding other rows.
- The **PROJECT** operation, selects certain **columns** from the relation and discards the other columns.
- If we are interested in only certain attributes of a relation, we can use the **PROJECT** operation to project the relation over these attributes only.
- Therefore, the result of the **PROJECT** operation can be visualized as a **vertical partition** of the relation into two relations:
- **Relation-1** contains the result of the operation, and **relation-2** contains the discarded columns.

Module – 2 Introduction to Relational Algebra

Unary Relational operations

The PROJECT Operation

The **general form** of the PROJECT operation is

$$\pi_{<\text{attribute list}>}(R)$$

Where **π (pi)** is the symbol used to represent the PROJECT operation.

<**attribute list**>is the desired sub-list of attributes from the attributes of relation **R**.

- The **result of the PROJECT** operation has only the attributes specified in <**attribute list**> in the same order as they appear in the list.
- Hence, its **degree** is equal to the number of attributes in <**attribute list**> .

Module – 2 Introduction to Relational Algebra

Unary Relational operations

The PROJECT Operation

- If the attribute list includes only non-key attributes of R, duplicate tuples are likely to occur.
- The **PROJECT** operation **removes any duplicate tuples**, so the result of the PROJECT operation is a **set of distinct tuples**, and hence a **valid relation**.
- This is known as **duplicate elimination**.
- Duplicate elimination **involves sorting** or some **other technique** to detect duplicates and thus adds more processing.
- **For example**, consider the following PROJECT operation:

$$\pi_{Sex,\ Salary}(EMPLOYEE)$$

Module – 2 Introduction to Relational Algebra

Unary Relational operations

The PROJECT Operation

Output relation

Sex	Salary
M	30000
M	40000
F	25000
F	43000
M	38000
M	25000
M	55000

Equivalent SQL query

SELECT DISTINCT Sex, Salary FROM EMPLOYEE

- The number of tuples in a relation resulting from a PROJECT operation is always less than or equal to the number of tuples in R.
- If the projection list is a **super-key** of **R**, then the resulting relation has the **same number of tuples** as **R**.

Module – 2 Introduction to Relational Algebra

Unary Relational operations

The PROJECT Operation

$\pi_{Lname,\ Fname,\ Salary}(EMPLOYEE)$

Output relation

Lname	Fname	Salary
Smith	John	30000
Wong	Franklin	40000
Zelaya	Alicia	25000
Wallace	Jennifer	43000
Narayan	Ramesh	38000
English	Joyce	25000
Jabbar	Ahmad	25000
Borg	James	55000

Equivalent SQL query

SELECT DISTINCT Lname, Fname, Salary FROM EMPLOYEE

Note: commutative property does not hold on PROJECT operation.

Module – 2 Introduction to Relational Algebra

Sequences of Operations and the RENAME Operation

- For most queries, we need to apply several relational algebra operations one after the other.
- Either we can write the operations as a **single relational algebra expression** by nesting the operations as written below

$\pi_{Fname, Lname, Salary}(\sigma_{Dno=5}(EMPLOYEE))$ known as an **in-line expression.**

or

- we can apply **one operation at a time** and **create intermediate result relations**, then **apply other operation on intermediate result** using assignment operation, denoted by ← .

$DEP5\_EMPS \leftarrow \sigma_{Dno=5}(EMPLOYEE)$

$RESULT \leftarrow \pi_{Fname, Lname, Salary}(DEP5\_EMPS)$

Module – 2 Introduction to Relational Algebra

Sequences of Operations and the RENAME Operation

Output relation

Fname	Lname	Salary
John	Smith	30000
Franklin	Wong	40000
Ramesh	Narayan	38000
Joyce	English	25000

$$\pi_{Fname,\ Lname,\ Salary}(\sigma_{Dno=5}(EMPLOYEE))$$

Or

$$DEP5\_EMPS \leftarrow \sigma_{Dno=5}(EMPLOYEE)$$

$$RESULT \leftarrow \pi_{Fname,\ Lname,\ Salary}(DEP5\_EMPS)$$

The RENAME Operation

- We can also use this technique to rename the attributes in the intermediate and result relations.
- This can be useful in connection with more complex operations such as UNION and JOIN.

Module – 2 Introduction to Relational Algebra

Sequences of Operations and the RENAME Operation

- To rename the attributes in a relation, we simply list the **new attribute names in parentheses**.

$$TEMP \leftarrow \sigma_{Dno=5}(EMPLOYEE)$$

$$R_{(First\_name,\ Last\_name,\ Salary)} \leftarrow \pi_{Fname,\ Lname,\ Salary}(TEMP)$$

TEMP

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
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

R

First_name	Last_name	Salary
John	Smith	30000
Franklin	Wong	40000
Ramesh	Narayan	38000
Joyce	English	25000

Equivalent SQL query

```
SELECT E.Fname AS First_name, E.Lname AS Last_name, E.Salary AS Salary FROM EMPLOYEE AS E WHERE E.Dno=5;
```

Module – 2 Introduction to Relational Algebra

Relational Algebra Operations from Set Theory(Binary relational operations) UNION, INTERSECTION, and MINUS(SET DIFFERENCE**)**

UNION: The result of this operation, denoted by R ∪ S, is a relation that includes all tuples that are either in **R** or in **S** or in both **R** and **S**. **Duplicate tuples are eliminated.**

INTERSECTION: The result of this operation, denoted by R ∩ S, is a relation that includes all tuples that are in both **R** and **S**.

SET DIFFERENCE (or MINUS): The result of this operation, denoted by R – S, is a relation that includes all tuples that are in **R** but not in **S.**

These operations are called as **binary relational operations**; that is, each is applied to two sets (of tuples).

Module – 2 Introduction to Relational Algebra

Binary relational operations

The UNION, INTERSECTION, and MINUS(SET DIFFERENCE) Operations

- When any of these three operations are applied on two relations, these two relations must have the same type of tuples; this condition has been called **union compatibility** or **type compatibility**.
- Two relations $R(A_1, A_2, \ldots, A_n)$ and $S(B_1, B_2, \ldots, B_n)$ are said to be **union compatible** (or type compatible) if they have the **same degree n** and if $dom(A_i) = dom(B_i)$ for $1 \leq i \leq n$.
- **This means** that the two relations have the same number of attributes and each corresponding pair of attributes has the same domain.

Module – 2 Introduction to Relational Algebra

Binary relational operations

Example: retrieve the Social Security numbers of all employees who either **work in department 5 or directly supervise an employee who works in department 5**, we can use the UNION operation as follows:

DEP5_EMPS ← $\sigma_{Dno=5}$(EMPLOYEE)

RESULT1 ← π_{Ssn}(DEP5_EMPS)

RESULT2(Ssn) ← π_{Super_ssn}(DEP5_EMPS)

RESULT ← RESULT1 U RESULT2

DEP5_EMPS

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
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

Module – 2 Introduction to Relational Algebra

Binary relational operations

RESULT1 ← π_{Ssn}(DEP5_EMPS)

RESULT1

Ssn
123456789
333445555
666884444
453453453

RESULT2(Ssn) ← π_{Super_ssn}(DEP5_EMPS)

RESULT2

Ssn
333445555
888665555

RESULT ← RESULT1 U RESULT2

RESULT

Ssn
123456789
333445555
666884444
453453453
888665555

The Social Security numbers of all employees who either **work in department 5 or directly supervise an employee who works in department 5**.

Module – 2 Introduction to Relational Algebra

Binary relational operations

Two union-compatible or type-compatible relations

STUDENT

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR

Fname	Lname
John	Smith
Ricardo	Browne
Susan	Yao
Francis	Johnson
Ramesh	Shah

STUDENT ∪ INSTRUCTOR

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert
John	Smith
Ricardo	Browne
Francis	Johnson

STUDENT ∩ INSTRUCTOR

Fn	Ln
Susan	Yao
Ramesh	Shah

Module – 2 Introduction to Relational Algebra

Binary relational operations

STUDENT – INSTRUCTOR

Fn	Ln
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR - STUDENT

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

- **UNION** and **INTERSECTION** are **commutative** operations.

$$R \cup S = S \cup R$$

$$R \cap S = S \cap R$$

Module – 2 Introduction to Relational Algebra

Binary relational operations

- Both UNION and INTERSECTION can be treated as n-ary operations applicable to any number of relations because both are also **associative operations**; that is,

$$R \cup (S \cup T) = (R \cup S) \cup T$$

$$(R \cap S) \cap T = R \cap (S \cap T)$$

- The MINUS operation is not commutative; that is,

$$R - S \neq S - R$$

INTERSECTION can be expressed in terms of union and set difference as follows:

$$R \cap S = ((R \cup S) - (R - S)) - (S - R)$$

Module – 2 Introduction to Relational Algebra

Binary relational operations

Problem

TABLE T1

P	Q	R
10	a	5
15	b	8
25	a	6

TABLE T2

A	B	C
10	b	6
25	c	3
10	b	5

Write the resultant relations of the following operations.

- T1 ∪ T2
- T1 ∩T2
- T1 – T2
- T2 – T1

Module – 2 Introduction to Relational Algebra

Binary relational operations

The JOIN Operations

- The JOIN operation, denoted by ⋈ , is used to combine related tuples from two relations into single "longer" tuples.
- The general form of a JOIN operation on two relations R(A_1, A_2, … , A_n) and S(B_1, B_2, … , B_m) is

$$R \bowtie_{\text{<join condition>}} S$$

- The result of the JOIN is a relation Q with n + m attributes Q(A_1, A_2, … , A_n, B_1, B_2, … , B_m) in that order; Q has one tuple for each combination of tuples, one from R and one from S, whenever the combination satisfies the join condition.

Module – 2 Introduction to Relational Algebra

Binary relational operations

The JOIN Operation

Example: retrieve the name of the department managers.

- To get the manager‘s name, we need to combine each department tuple with the employee tuple whose Ssn in EMPLOYEE value matches the Mgr_ssn value in the department tuple.
- This can be accomplished by using the JOIN operation and then projecting the result over the necessary attributes, as follows:

$$DEPT\_MGR \leftarrow DEPARTMENT \bowtie_{Mgr\_ssn=Ssn} EMPLOYEE$$

DEPT_MGR

Dname	Dnumber	Mgr_ssn	...	Fname	Minit	Lname	Ssn	...
Research	5	333445555	...	Franklin	T	Wong	333445555	...
Administration	4	987654321	...	Jennifer	S	Wallace	987654321	...
Headquarters	1	888665555	...	James	E	Borg	888665555	...

Module – 2 Introduction to Relational Algebra

Binary relational operations

Variations of JOIN: The EQUIJOIN and NATURAL JOIN

- The most common use of JOIN involves join conditions with equality comparisons only. Such a JOIN, where the only comparison operator used is =, is called an **EQUIJOIN**.
- In the result of an EQUIJOIN we always have one or more pairs of attributes that have identical values in every tuple.

Example

superfluous

$$DEPT\_MGR \leftarrow DEPARTMENT \bowtie_{Mgr\_ssn=Ssn} EMPLOYEE$$

DEPT_MGR

Dname	Dnumber	Mgr_ssn	...	Fname	Minit	Lname	Ssn	...
Research	5	333445555	...	Franklin	T	Wong	333445555	...
Administration	4	987654321	...	Jennifer	S	Wallace	987654321	...
Headquarters	1	888665555	...	James	E	Borg	888665555	...

Module – 2 Introduction to Relational Algebra

Binary relational operations

- A new operation called NATURAL JOIN—denoted by * was created to get rid of the second (superfluous) attribute in an EQUIJOIN condition.
- The **standard definition of NATURAL JOIN** requires that the two join attributes have the same name in both relations. If this is not the case, a renaming operation is applied first.

$$PROJ\_DEPT \leftarrow PROJECT * \rho_{(Dname,\ Dnum,\ Mgr\_ssn,\ Mgr\_start\_date)}(DEPARTMENT)$$

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

Module – 2 Introduction to Relational Algebra

Binary relational operations

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

Note: if no combination of tuples satisfies the join condition, the result of a JOIN is an empty relation with zero tuples.

PROJ_DEPT

Pname	Pnumber	Plocation	Dnum	Dname	Mgr_ssn	Mgr_start_date
ProductX	1	Bellaire	5	Research	333445555	1988-05-22
ProductY	2	Sugarland	5	Research	333445555	1988-05-22
ProductZ	3	Houston	5	Research	333445555	1988-05-22
Computerization	10	Stafford	4	Administration	987654321	1995-01-01
Reorganization	20	Houston	1	Headquarters	888665555	1981-06-19
Newbenefits	30	Stafford	4	Administration	987654321	1995-01-01

Module – 2 Introduction to Relational Algebra

Binary relational operations

- The resultant relation of JOIN operations (Equijoin and Natural join) has tuples that satisfy the join condition. Hence, tuples which do not satisfy join condition are eliminated from the JOIN result. Tuples with NULL values in the join attributes are also eliminated. This type of join, is known as an **inner join**.
- A set of operations, called **outer joins**, were developed for the case where the user wants to keep all the tuples in R, or all those in S, or all those in both relations in the result of the JOIN, regardless of whether or not they have matching tuples in the other relation.

Types of outer joins

- Left outer join
- Right outer join

Module – 2 Introduction to Relational Algebra

Binary relational operations

Problem

TABLE T1

P	Q	R
10	a	5
15	b	8
25	a	6

TABLE T2

A	B	C
10	b	6
25	c	3
10	b	5

Write the resultant relations of the following operations.

- $T1 \bowtie_{T1.P = T2.A} T2$
- $T1 \bowtie_{T1.Q = T2.B} T2$
- $T1 \rtimes_{T1.P = T2.A} T2$
- $T1 \ltimes_{T1.Q = T2.B} T2$
- $T1 \bowtie_{(T1.P = T2.A \text{ AND } T1.R = T2.C)} T2$

Module – 2 Introduction to Relational Algebra

Additional Relational Operations

Aggregate Functions and Grouping

- Common functions applied to collections of numeric values include SUM, AVERAGE, MAXIMUM, and MINIMUM. The COUNT function is used for counting tuples or values.
- Another common type of request involves grouping the tuples in a relation by the value of some of their attributes and then applying an aggregate function independently to each group.

General form

$$\langle \text{grouping attributes} \rangle \, \mathfrak{I} \, \langle \text{function list} \rangle \, (R)$$

where <grouping attributes> is a list of attributes of the relation specified in R, and <function list> is a list of (<function><attribute>) pairs.

Module – 2 Introduction to Relational Algebra

Additional Relational Operations

Problem: write a algebraic expression to retrieve each department number, the number of employees in the department, and their average salary.

Solution:

$$\rho_{R(\text{Dno, No\_of\_employees, Average\_sal})} (_{\text{Dno}}\Im_{\text{COUNT Ssn, AVERAGE Salary}} (\text{EMPLOYEE}))$$

R

Dno	No_of_employees	Average_sal
5	4	33250
4	3	31000
1	1	55000

Module – 2 Introduction to Relational Algebra

- If no grouping attributes are specified, the functions are applied to all the tuples in the relation, so the resulting relation has a single tuple only.

Example:

$$\Im_{\text{COUNT Ssn, AVERAGE Salary}}(\text{EMPLOYEE}).$$

Count_ssn	Average_salary
8	35125

Module – 2 Introduction to Relational Algebra

Source

Text books

1. Fundamentals of Database Systems, Ramez Elmasri and Shamkant B. Navathe, 7th Edition, 2017, Pearson.
2. Database management systems, Ramakrishnan, and Gehrke, 3rd Edition, 2014, McGraw Hill