Fpt University

Chapter 2 The Relational Model of Data

Objectives

- Understand what is the relational model and database design basing relational model.
- Conceptualize data using the relational model.
- Understand what basic relational algebra operators under set semantics.
- Express queries using relational algebra.

Contents

2.1 An Overview of Data Models

- **Data model**: a collection of concepts for describing data, including 3 parts:
  - Structure of the data

    - Ex: arrays or objects
  - Operations on the data

    - Queries and modification on data
  - Constraints on the data

    - Limitations on the data

2.1 An Overview of Data Models

- What Is a Data Model?

A data model is a diagram of how data is organized and stored and the relationships between that information.

- A data model identifies the data, the data attributes, and the relationships or associations with other data.

- A data model provides a generalized the real business scenario

Example: car store data model

- Car: Make, year of manufacture, color and size of the car

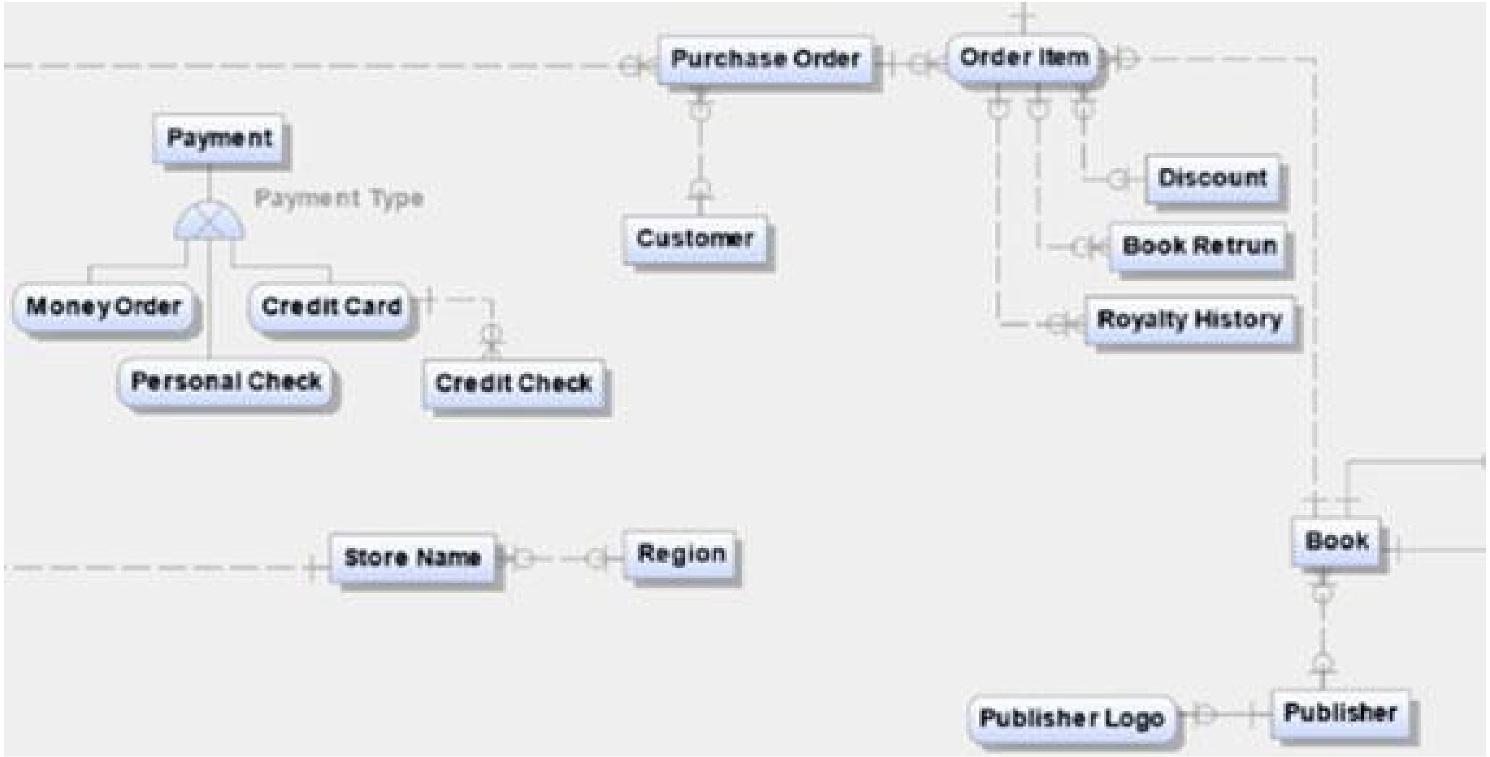
- Customers: full name, ID card, phone number

- The relationship is: Buy (date, quantity, into money...)

2.1 An Overview of Data Models

❑What are the different types of data models?

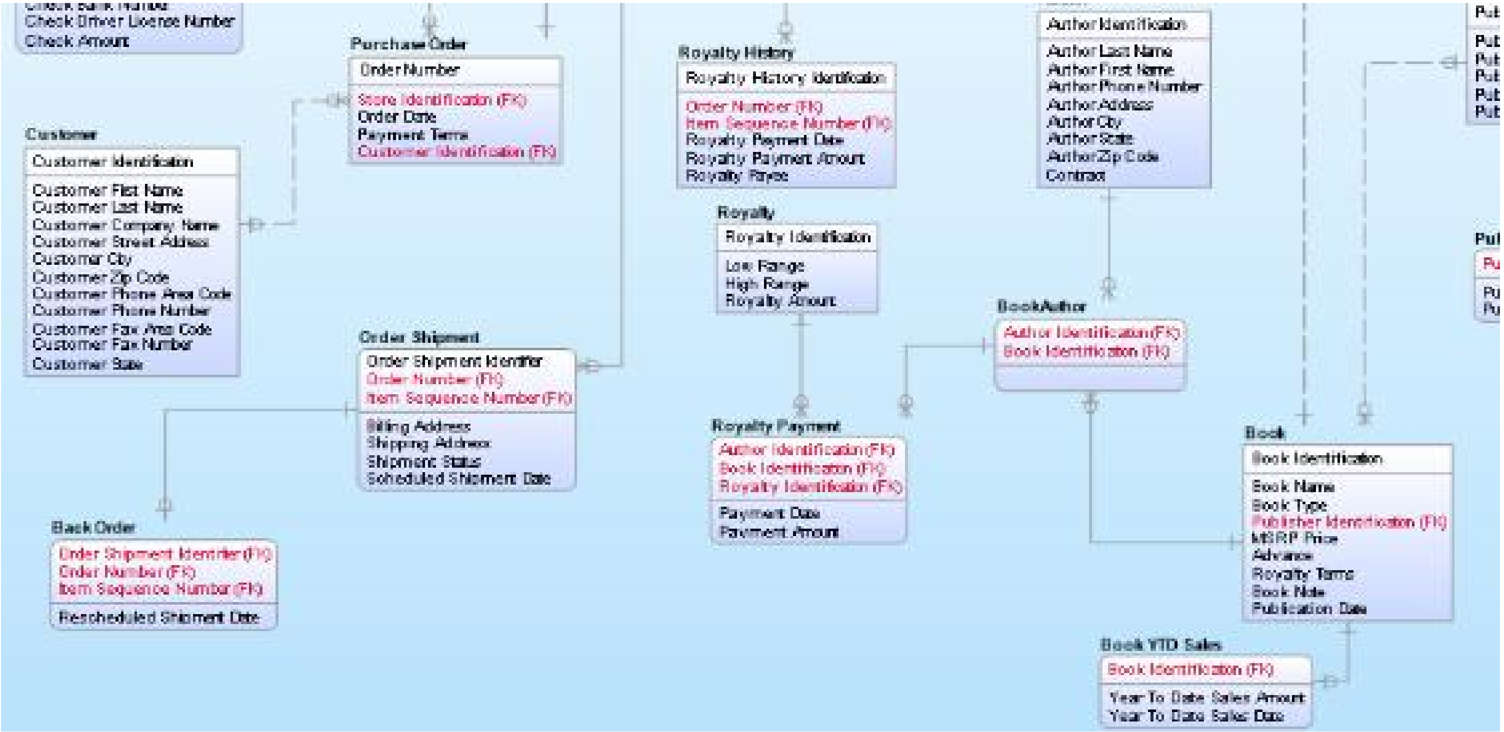
- Conceptual data models



2.1 An Overview of Data Models

- Logical data models

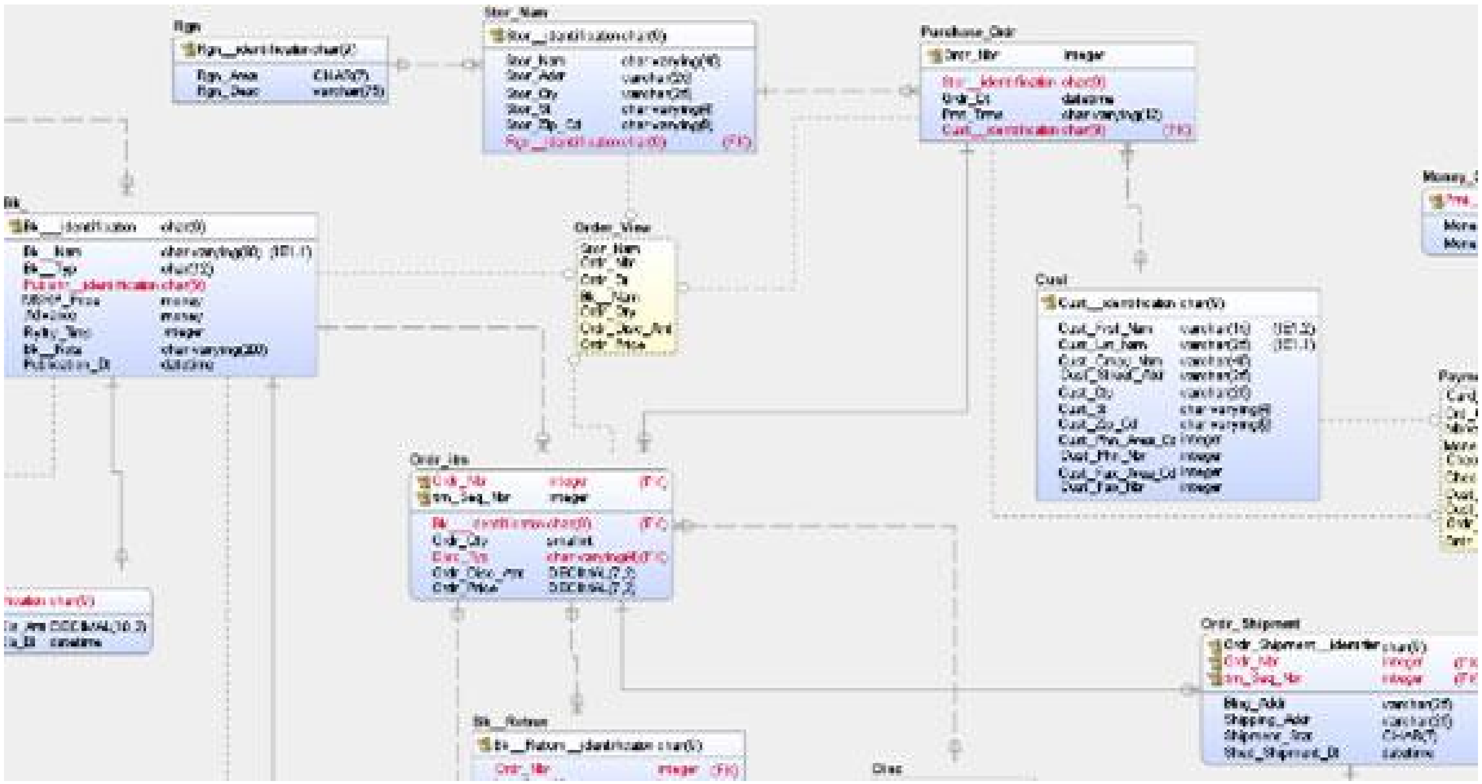
A logical data model has three main components: entities, relationships and attributes



2.1 An Overview of Data Models

- Physical data models

The includes all of the various tables, the columns on those tables and the relationships between them.



2.2 Basics of the Relational Model

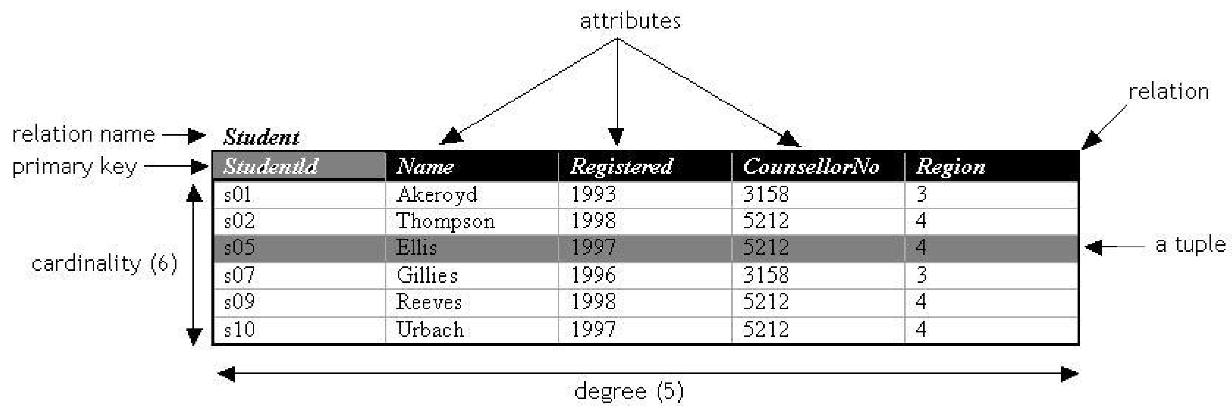
- **Relational model**
  - A relation is made up from 2 parts:

    - Schema: specifies name of relation, name of attributes and domain/type of one's.

    - Ex: Student(StudentID: string, Name: string, Registered: int, CounsellorNo: int, Region: int)

    - a table with rows and columns

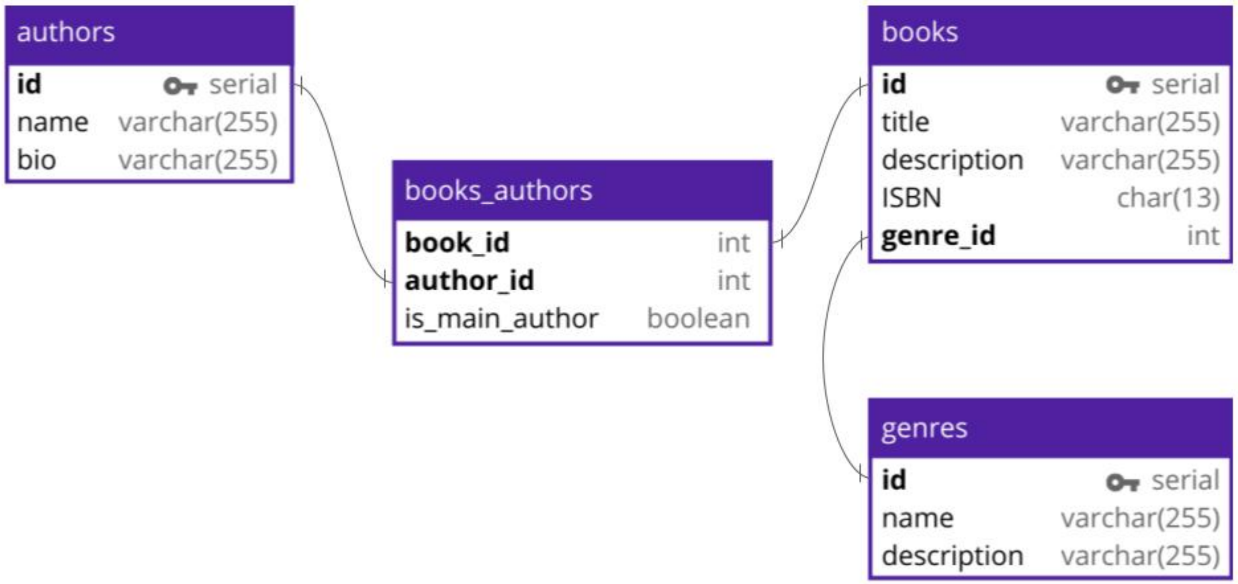
    - Rows ~ cardinality; columns ~ degree/arity
  - **A simple thinking: a relation as a set of distinct rows or tuples**



StudentId	Name	Registered	CounsellorNo	Region
s01	Akeroyd	1993	3158	3
s02	Thompson	1998	5212	4
s05	Ellis	1997	5212	4
s07	Gillies	1996	3158	3
s09	Reeves	1998	5212	4
s10	Urbach	1997	5212	4

2.2 Basics of the Relational Model

- Database schema: a set of schemas for the relations of a database
- An example of DB schema:



Fpt University

2.2 Basics of the Relational Model

- Key attribute
- Non-key attribute
- Multi-valued attribute
- Derived- attribute
- Candidate key
- Primary key
- Foreign key

2.2 Basics of the Relational Model

▪Keys of relation

Each row has one or more attributes, known as relation key, which can identify the row in the relation (table) uniquely.

- A set of attributes forms a key → don't allow 2 tuples in a relation instance to have the same values

Ex: CUSTOMER (**Customer ID**, Name, Name, Address)

→ CustomerID

Customer ID	Name	Address
1	Thuyen	20 Hoang Hoa Tham
2	Thuyen	50 Nguyen Thai Hoc
3	Tuan	20 Chu Van An

2.2 Basics of the Relational Model

- **Composite Key**

 - Sometimes, a single column/attribute not have uniquely identifies all the records of a table

- To uniquely identify rows of a table, a combination of two or more columns/attributes can be used.

--> It acts as a primary key if there is no primary key in a table

-->Two or more attributes are used together to make a composite key.

2.2 Basics of the Relational Model

STUDENT

Composite Key

S_Name	S_Class	Parent_Contact_No	S_Age
Mehul	6	8700867330	11
Rashi	5	8700867330	10
Mehul	6	9990155289	11
Vansh	7	9354226009	12
[illegible]	[illegible]	868101221	[illegible]

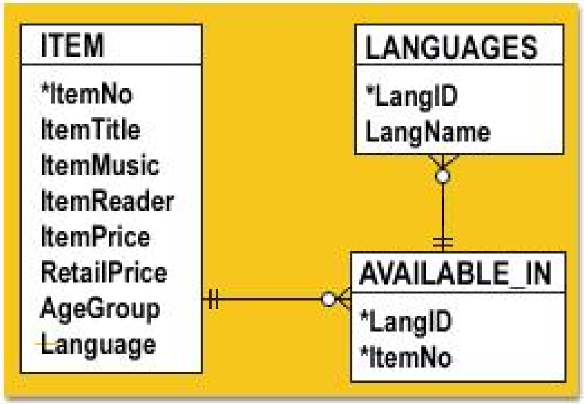
2.2 Basics of the Relational Model

Why do we use key?

- Keys help identify any row of data in the table --> identification of a single record in the database.
- Allows setting and defining relationships between tables
- Helps enforce uniqueness and integrity in relationships.

2.2 Basics of the Relational Model

- **Key attribute:** The Key attribute is used to denote the property that uniquely identifies an entity (the EntityKey). such as size, shape, weight, and color, etc.,



- **Non-key attribute:** Non-key attributes are attributes that are not part of any key. Generally, most attributes are simply descriptive

EX: Class_name, Room, Age, … → are the non-key attributes.

2.2 Basics of the Relational Model

- **Multi-valued attribute**

- **A single-valued attribute** is an attribute where each instance of an entity can have only one value.

Ex:

- The attributes of an Employee entity: id, birthday, Gender

- An employee has only one employee id which is unique and it also has a single date of birth.

→ So these attributes can store only one value in it. Therefore, it is known as Single Valued Attributes.

2.2 Basics of the Relational Model

- **Multi-valued attribute**

- **A multivalued attribute** is an attribute where each instance of an entity can take on more than one value.

- EX:

1. Expertise is a multivalued attribute of the TEACHER: 'Programming', 'Advanced Math' and 'Database'….etc

2. Email id and phone: can provide more than one email id and contact no.

→ Therefore multiple values can be stored in Multi-Valued Attributes

2.2 Basics of the Relational Model

- **Derived- attribute**

- Derived properties are calculated from values that are previously stored values.

- Stored attributes help calculate the value of derived attributes.

For example, **date of birth is an attribute** with the help of which we can calculate the **age of the person**.

If you find the age column in the table then it will be called a derived attribute because they are mainly calculated from a stored property like date of birth.

2.2 Basics of the Relational Model

- **Candidate key**: The minimal set of attributes that can uniquely identify a tuple is known as a candidate key.

EX: STUD_NO is Candidate key STUDENT

- It must contain unique values

- It can contain NULL values

- Every table must have at least a single candidate key.

- A table can have multiple candidate keys but only one primary key (the primary key cannot have a NULL value, so the candidate key with a NULL value can't be the primary key).

- There can be more than one candidate key in a relationship.

2.2 Basics of the Relational Model

The candidate key can be simple (having only one attribute) or composite as well

Table STUDENT

STUD_NO	SNAME	ADDRESS	PHONE
1	Shyam	Delhi	123456789
2	Rakesh	Kolkata	223365796
3	Suraj	Delhi	175468965

Table STUDENT_COURSE

STUD_NO	TEACHER_NO	COURSE_NO
1	001	C001
2	056	C005

```
+----+---------+--------+--------+--------------------+--------+
| Id | Name    | Gender | City   | Email              | Dep_Id |
+----+---------+--------+--------+--------------------+--------+
| 1  | Ajay    | M      | Delhi  | ajay@gmail.com     |      1 |
| 2  | Vijay   | M      | Mumbai | vijay@gmail.com    |      2 |
| 3  | Radhika | F      | Bhopal | radhika@gmail.com  |      1 |
| 4  | Shikha  | F      | Jaipur | shikha@gmail.com   |      2 |
| 5  | Hritik  | M      | Jaipur | hritik@gmail.com   |      2 |
+----+---------+--------+--------+--------------------+--------+
5 rows in set (0.00 sec)
```

2.2 Basics of the Relational Model

- **Primary key**

The primary key is used to uniquely identify each table in the database table.

- Used to establish a relationship (or reference constraint) between two tables in the database.

- The data of the primary key field must be unique. And does not contain Null values.

-> The **choice of a primary key** in a relational database often depends on the preference of the **administrator**.

2.2 Basics of the Relational Model

- **Foreign key**

- If an attribute can only take the values which are present as values of some other attribute, it will be a foreign key to the attribute to which it refers.

- The referenced attribute of the referenced relation should be the primary key to it.

- It is a key it acts as a primary key in one table and it acts as secondary key in another table.

- It combines two or more relations (tables) at a time.

- They act as a cross-reference between the tables.

2.2 Basics of the Relational Model

Primary Key

Foreign Key

ID	Name	Course
2041	Tom	Java
2204	John	C++
2043	Alice	Python
2032	Bob	Oracle

Student Details

ID	Marks
2041	65
2204	55
2043	73
2032	62

Student Marks

2.2 Basics of the Relational Model

idStudent	Name	Date	idClass	Phone
K17011	Thuyen	20/10/2003	K17AI	072456789
K17022	Tuan	10/11/2003	K17SE	1546789542
K17023	Hoa	11/11/2002	K17AI	564654
K17024	Nghia	5/1/2002	K17SE	45666

Foreign key

N_Order	idStudent	Subject	Summer22	Fall22	Spring23
1	K17011	CR250			
2	K17001	CS366			
3	K17022	CR250			
4	K17024	CS366			

2.3 An Algebraic Query Language

Relational Algebra

- An algebra consists of operators and atomic operands
- Relational algebra is an example of an algebra, its atomic operands are
  - Variables that stand for relations
  - Constants, which are finite relations
- Relational algebra is a set of operations on relations
- Operations operate on one or more relations to create new relation

2.3 An Algebraic Query Language

Relational algebra fall into four classes

- Set operations – union, intersection, difference
- Selection and projection
- Cartesian product and joins
- Rename

2.3 An Algebraic Query Language

- **Set operations**

- Union

$$R \cup S = \{ t \mid t \in R \lor t \in S \}$$

- Intersection

$$R \cap S = \{ t \mid t \in R \land t \in S \}$$

- Difference

$$R \setminus S = \{ t \mid t \in R \land t \notin S \}$$

- Intersection can be expressed in terms of set difference

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

R and S must be 'type compatible'

- The same number of attributes
- The domain of corresponding attributes must be compatible

Set operations- Example

name	*address*	*gender*	*birthdate*
Carrie Fisher	123 Maple St., Holywood	F	9/9/99
Mark Hamill	456 Oak Rd., Brentwood	M	8/8/88

Relation R

name	*address*	*gender*	*birthdate*
Carrie Fisher	123 Maple St., Holywood	F	9/9/99
Harrison Ford	789 Palm Dr., Beverly Hills	M	8/8/88

Relation S

Fpt University

Set operations- Example

$R \cup S$

name	address	gender	birthdate
Carrie Fisher	123 Maple St., Holywood	F	9/9/99
Mark Hamill	456 Oak Rd., Brentwood	M	8/8/88
Harrison Ford	789 Palm Dr., Beverly Hills	M	8/8/88

$R \cap S$

name	address	gender	birthdate
Carrie Fisher	123 Maple St., Holywood	F	9/9/99

$R \setminus S$

name	address	gender	birthdate
Mark Hamill	456 Oak Rd., Brentwood	M	8/8/88

Selection and projection

- **Selection:** Selection is the selection of a subset of tuples of the given relation R, satisfying the given conditional expression

  - R1 := σ_C(R2) with C illustrated conditions

The selection on R2 under the condition C, symbol σ_C(R2) , results in a relation (R1) consisting of tuples of R2 such that the condition C is TRUE or false (FALSE).

$$\mathbf{R1 := \sigma_C(R2) = \{ u \mid u \in R2 \text{ và } C(u) = TRUE \}}$$

Selection and projection

-The logical operations (logical) in the expression F: NOT (¬), AND (∩), OR (∪)

- Comparison operations in the expression F: =, ≠, <, ≤, >, ≥

- The result is a new relation with the same list of attributes as r, and whose tuples are always less than or equal to the number of tuples of r.

- The selection operation is unary, that is, it is applied to a relation.

- **The selection is commutative**: $\sigma_{C1}(\sigma_{C\,2}(R2)) = \sigma_{C2}(\sigma_{C\,1}(R2)) = \sigma_{C1} \cap \sigma_{C2}(R2)$

→ We can combine a series of selections into a simple selection using the AND operator (∩).

For example: $\sigma_{C1}(\sigma_{C\,2}(R2)) =)) = \sigma_{C2\ and}\ \sigma_{C1}(R2)$

Selection and projection

Example for a relationship: Student

ID	fullName	idCourse	Dep	marks
99001	Tran Van Thu	Database	IT	3.0
99002	Nguyen Da Thao	Database	IT	8.0
99001	Tran Van Thu	CSI104	IT	6.0
99005	Le Van Trung	CSI104	Language	5.0

? Find database students with test scores above 5.0

σ(idCourse = Database ∩ marks > 5)(Student)

Result

ID	fullName	idCourse	Dep	marks
99002	Nguyen Da Thao	Database	IT	8.0

Selection and projection

Movies

title	*year*	*length*	*genre*
Gone With the Wind	1939	231	Drama
Star Wars	1977	124	Scifi
Wayne’s World	1992	95	Comedy

$\sigma_{length \geq 100}(\text{Movies})$

title	*year*	*length*	*genre*
Gone With the Wind	1939	231	Drama
Star Wars	1977	124	Scifi

Selection and projection

- **Projection** Projection on a relation is essentially the removal of some properties of that relation.

-Given a relation schema R(A1, A2, …, An) and a subset of attributes X, with X $\subset$ {A1, A2,…, An}.

-Call t a tuple of R, A an attribute, t[A] the value of tuple t at attribute A.

-We have X = { B1, B2, …, Bm }. Then, t[X] is the value of tuple t on the attribute set X, t[X] = { t[B1], t[B2], …, t[Bm] }

→The projection r(R) on an attribute set X, $\pi_x(r)$ is defined as follows: $\pi_x(r) = \{ t[X] / t \in r \}$

Selection and projection

The projection of the relation R on the attribute set X is a set of tuples, → Removing from the t tuples in the relation R those attributes that are not in X.

--> The return result: a relation with m attributes in X and in the same order as their order in the R

--> The number of result sets : always less than or equal to the number of tuples in R.

Selection and projection

Movies

title	*year*	*length*	*genre*
Star Wars	1977	124	Scifi
Galaxy Quest	1999	104	Comedy
Wayne's World	1992	95	Comedy

$\pi_{title,year,length}$(Movies)

title	*year*	*length*
Star Wars	1977	124
Galaxy Quest	1999	104
Wayne's World	1992	95

π_{genre}(Movies)

genre
Scifi
Comedy

Cartesian product and joins

Cartesian product: R3 := R1 X R2

- Let the relation r on the relation schema R (A1, A2, …, Am)
- And s on the relation schema S (B1, B2,…, Bn).

The Cartesian product of two relations r and s, denoted r × s → is a relation on the schema T(A1, A2, …, Am, B1, B2, …, Bn) consisting of tuples u such that m the first component is a tuple of r and the last n component is a tuple of s.

r × s = { (u1, …, um, um+1, …, um+n) | (u1, …, um) ∈ r và (um+1, …,um+n) ∈ s}

Cartesian product and joins

Attribute B of the same name in R and S: use R.B and S.B

Relation R

A	B
1	2
3	4

Relation S

B	C	D
2	5	6
4	7	8
9	10	11

Cartesian Product R X S

A	R.B	S.B	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

Cartesian product and joins

Relation R

A	B	C
1	2	3
4	5	6
1	2	7
8	4	5

Relation S

T	D
1	5
3	7

Cartesian Product R X S

A	B	C	T	D
1	2	3	1	5
1	2	3	3	7
4	5	6	1	5
4	5	6	3	7
1	2	7	1	5
1	2	7	3	7
8	4	5	1	5
8	4	5	3	7

Cartesian product and joins

joins

- Join is used to combine related tuples from two relations into a tuple.
- The general form of the union on two relations R(A1, A2,…,An) and S(B1, B2, …, Bm) is:

$Q := R \bowtie_{\langle \text{join condition} \rangle} S$

- The result of the join is a relation Q(A1, A2, …,An, B1, B2, …, Bm) with n+m attributes.
- Each tuple of Q is a connection between a tuple of R and a tuple of S → when they satisfy the join condition.

Cartesian product and joins

- joins

- **The difference between the Cartesian product and the join:**

+ Join: only tuples that satisfy the termination condition appear in the result

+ Cartesian product, all combinations of tuples are equal→ included in the results.

- The join condition is specified on the attributes of the two relations R and S:

<Condition> AND <Condition> AND … AND <Condition>

Cartesian product and joins

- **joins**

Relation R

A	B	C
1	2	3
4	5	6
1	2	7
8	4	5

Relation S

A	D
1	5
3	7

Result : $R \bowtie_{R.A>S.A} S$

R.A	B	C	S.A	D
4	5	6	1	5
4	5	6	3	7
8	4	5	1	5
8	4	5	3	7

Cartesian product and joins

- joins

A	B	C
1	2	3
6	7	8
9	7	8

Relation U

B	C	D
2	3	4
2	3	5
7	8	10

Relation V

A	U.B	U.C	V.B	V.C	D
1	2	3	2	3	4
1	2	3	2	3	5
1	2	3	7	8	10
6	7	8	7	8	10
9	7	8	7	8	10

Result of $U \bowtie_{A<D} V$

A	U.B	U.C	V.B	V.C	D
1	2	3	7	8	10

Result of $U \bowtie_{A<D \text{ AND } U.B \neq V.B} V$

Cartesian product and joins

Inner join (equivalent join)

An internal join is a join with the condition that there is an equality (=) between the primary key and the foreign key.

R $_{\text{<R.Primary Key = S.Foreign Key>}}$ S

Student

ID	fullName	Dep
99001	Tran Van Thu	IT
99002	Nguyen Da Thao	IT
99001	Tran Van Thu	IT
99005	Le Van Trung	Language

Faculty

Dep	Faculty name
IT	Information Technology
IT	Information Technology
IT	Information Technology
Language	English language

Cartesian product and joins

$\text{Student} \bowtie_{\text{Student.Dep = Faculty.Dep}} \text{Faculty}$

ID	fullName	Student.Dep	Faculty.Dep	Faculty name
99001	Tran Van Thu	IT	IT	Information Technology
99002	Nguyen Da Thao	IT	IT	Information Technology
99001	Tran Van Thu	IT	IT	Information Technology
99005	Le Van Trung	Language	Language	English language

Cartesian product and joins

Natural join

$R3 := R1 \bowtie R2$

- Pair only those tuples from R1 and R2 that attributes are common to the schema of R1 and R2

Relation R

A	B
1	2
3	4

Relation S

B	C	D
2	5	6
4	7	8
9	10	11

Natural Join $R \bowtie S$

A	B	C	D
1	2	5	6
3	4	7	8

Rename

- The ρ operation gives a new schema to a relation
- $\rho_{S(A1,\ldots,An)}(R)$ makes S be a relation with attributes A1,…,A*n* and the same tuples as R
- Simplified notation: S:=R (A1,A2,…,An)

→When we are joining two or more tables and if those tables have the same column name, it's better to rename the columns to distinguish them

→ An operation used to rename the attributes of a relation.

Rename

- Rename the table

ρ Student (STD_TABLE) → Rename table STD_TABLE to STUDENT

- Rename table columns

STUDENT has columns: ID, NAME and ADDRESS --> Rename to: STD_ID, STD_NAME, STD_ADDRESS

ρ STD_ID, STD_NAME, STD_ADDRESS(STUDENT)

Rename

Relation R

A	B
1	2
3	4

Relation S

B	C	D
2	5	6
4	7	8
9	10	11

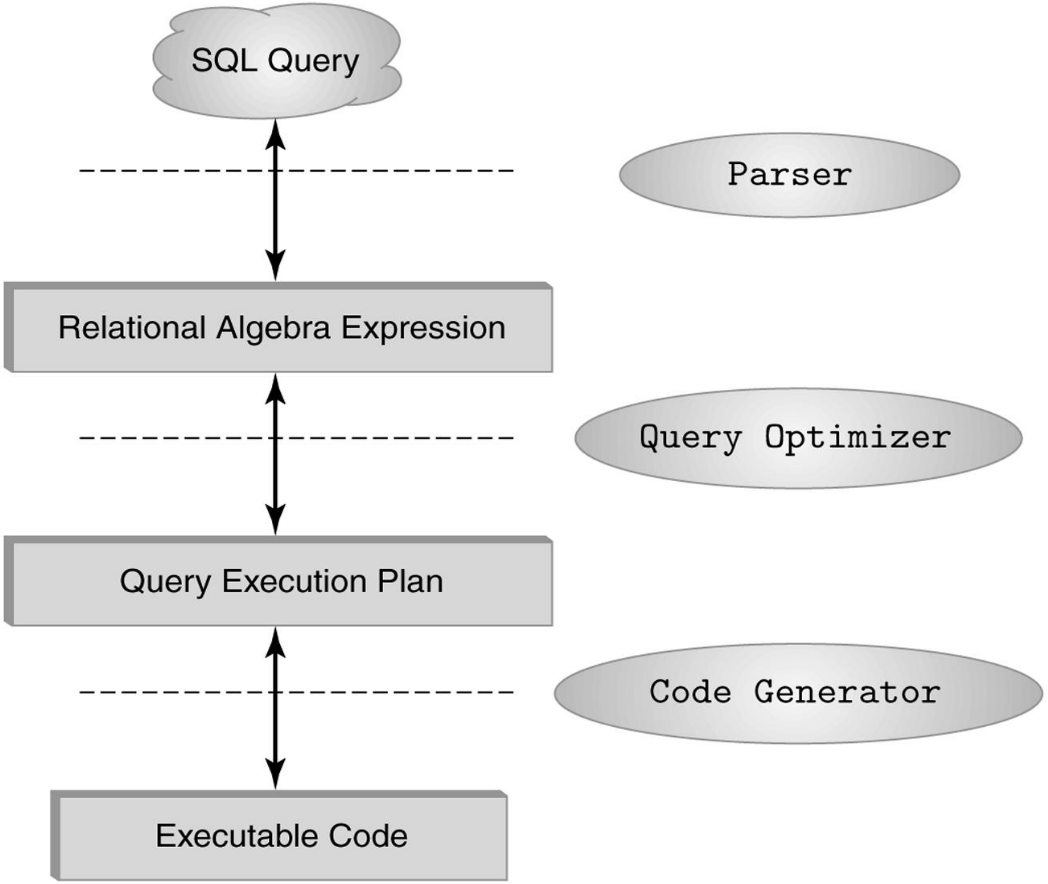
R X $\rho_{S(X,C,D)}$ (S)

A	B	X	C	D
1	2	2	5	6
1	2	4	7	8
1	2	9	10	11
3	4	2	5	6
3	4	4	7	8
3	4	9	10	11

Relational Expression

- How we need relational expression
- Relational algebra allows us to form expressions
- Relational expression is constructed by applying operations to the result of other operations
- Expressions can be presented as expression tree

The role of relational algebra in a DBMS



Relational Expression

Example: What are the titles and years of movies made by Fox that are at least 100 minutes long?

- (1) Select those Movies tuples that have length ≥ 100
- (2) Select those Movies tuples that have studioName='Fox'
- (3) Compute the intersection of (1) and (2)
- (4) Project the relation from (3) onto attributes title and year

Relational Expression

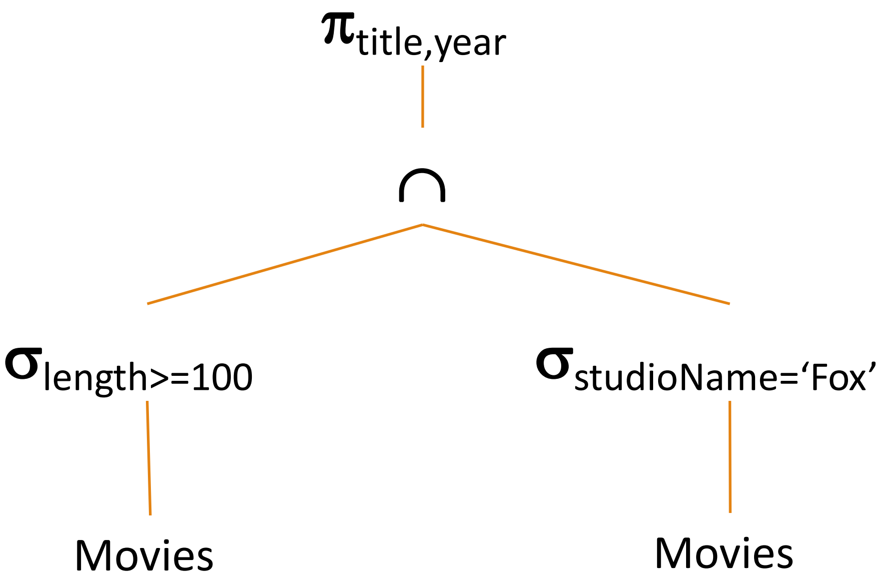


Figure 2.18: Expression tree for a relational algebra expression

$$\pi_{title,year}(\sigma_{length \geq 100}\,(Movies) \cap \sigma_{studioName='Fox'}\,(Movies))$$

$$\pi_{title,year}(\sigma_{length \geq 100 \text{ AND } studioName='Fox'}\,(Movies))$$

Relational Expression

DEPOSITOR

CUSTOMER_NAME	ACCOUNT_NO	CITY
Johnson	A-101	Harrison
Smith	A-121	Rye
Mayes	A-321	Harrison
Turner	A-176	Rye
Johnson	A-273	Brooklyn
Jones	A-472	Harrison
Lindsay	A-284	North

BORROW

BRANCH_NAME	CUSTOMER_NAME	LOAN_NO	AMOUNT
Downtown	Jones	L-17	1000
Redwood	Smith	L-23	2000
Perryride	Hayes	L-15	1500
Downtown	Jackson	L-14	1500
Mianus	Curry	L-13	500
Roundhill	Smith	L-11	900
Perryride	Williams	L-16	1300
Downtown	Johnson	L-18	500

Relational Expression

1. Find the branch with the name "**perryride**“
2. Display the **name and city** of the customer customers who are depositors
3. Displays the names of customers who have both **DEPOSITOR** and **borrowed**
4. Display all names of customers who **deposit** or **borrow**
5. Show all names of customers who only **DEPOSITOR** and don't **borrow**
6. Selects tuples from **borrow** where AMOUNT >=1500
7. Selects tuples from BORROW where BRANCH_NAME is 'Downtown' and AMOUNT > 1000 or LOAN_NO = ‘L-17’
8. Find information about borrowers and borrowers whose name is "Johnson"