四川大學
SICHUAN UNIVERSITY

Database System Concepts

Relational Model

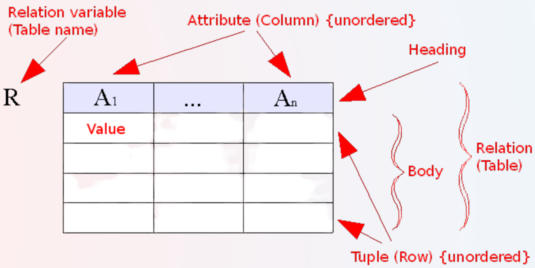
伍元凯

College of Computer Science (Software), Sichuan University

wuyk0@scu.edu.cn

2023/03/01

海纳百川
有容乃大



Relational model concepts

- the term relation（关系） is used to refer to a table(表)
- the term tuple（元组、记录） is used to refer to a row(行)
- the term attribute（属性） is used to refer to a column(列)

- The current values (relation instance) of a relation are specified by a table
- An element t of r is a tuple, represented by a row in a table

Relation instance for "Sales"

Sales_ID	Product_ID	Product_Name	Category	Quantity	Price
1	101	Product A	A	10	100
2	102	Product B	B	20	200
3	103	Product C	A	30	300

Order of tuples is irrelevant (tuples may be stored in an arbitrary order)

Customer ID	Customer Name	City
101	Alice	New York
102	Bob	Boston
103	Charlie	San Francisco

Customer ID	Customer Name	City
102	Bob	Boston
101	Alice	New York
103	Charlie	San Francisco

The set of allowed values for each attribute is called the domain（域） of the attribute

The special value null （空值） is a member of every domain

The null value causes complications in the definition of many operations

DOMAIN Salary $\in \mathbb{R}^+$.

```
CREATE TABLE Employees (
    Employee_ID INT PRIMARY KEY,
    First_Name VARCHAR(50),
    Last_Name VARCHAR(50),
    Hire_Date DATE,
    Salary DECIMAL(10,2),
    Department_ID INT,
    CONSTRAINT salary_range CHECK (Salary >= 0)
);
```

Attribute values are (normally) required to be atomic（原子的）; that is, indivisible

Table 1: Violating atomic

Customer ID	Order Details
101	Pen, Pencil
102	Notebook, Pen, Pencil

Employee			
Attribute	**Data Type**	**Domain**	**Description**
Employee_ID	INTEGER	$\mathbb{N}$	Unique identifier for each employee
First_Name	VARCHAR(50)	Σ^{50}	First name of the employee
Last_Name	VARCHAR(50)	Σ^{50}	Last name of the employee
Gender	CHAR(1)	{'M', 'F'}	Gender of the employee
Date_of_Birth	DATE	$\mathbb{D}$	Date of birth of the employee
Salary	DECIMAL(10,2)	$\mathbb{R}^{+}$	Salary of the employee

Employee Relation Schema

$Employee(Employee\_ID, First\_Name, Last\_Name, Gender, Date\_of\_Birth, Salary)$

Employee_ID	First_Name	Last_Name	Gender	Date_of_Birth	Salary
001	John	Smith	M	1990-02-08	100,000
002	Jane	Doe	F	1985-06-21	80,000
003	Bob	Johnson	M	1975-10-12	120,000
004	Alice	Lee	F	1995-03-15	70,000
005	Michael	Chen	M	1980-12-01	90,000

Employee Relation Instance

A superkey K is a **set** of attributes $A_1, A_2, ..., A_n$ such that no two tuples (rows) in a relation have the same values for all of the attributes in K. Allow us to identify uniquely a tuple in the relation.

ID	FirstName	LastName	Email	Phone
1	John	Smith	john.smith@example.com	555-1234
2	Jane	Doe	jane.doe@example.com	555-5678
3	Bob	Johnson	bob.johnson@example.com	555-9012
4	Alice	Williams	alice.williams@example.com	555-3456

Superkey: $\{ID\}$ and $\{ID, FirstName\}$
Superkey K is a candidate key(候选健) if K is minimal. Example: $\{ID\}$ is a candidate key for Instructor. One of the candidate keys is selected to be the primary key（主键）.

A set of relations with primary keys underlined:

Employees(<u>EmployeeID</u>, FirstName, LastName, HireDate, Address)

Departments(<u>DepartmentID</u>, DepartmentName, ManagerID)

Projects(<u>ProjectID</u>, ProjectName, Budget)

EmployeeProjects(<u>EmployeeID</u>, <u>ProjectID</u>, HoursWorked)

Address should not be the primary key, since it is likely to change.

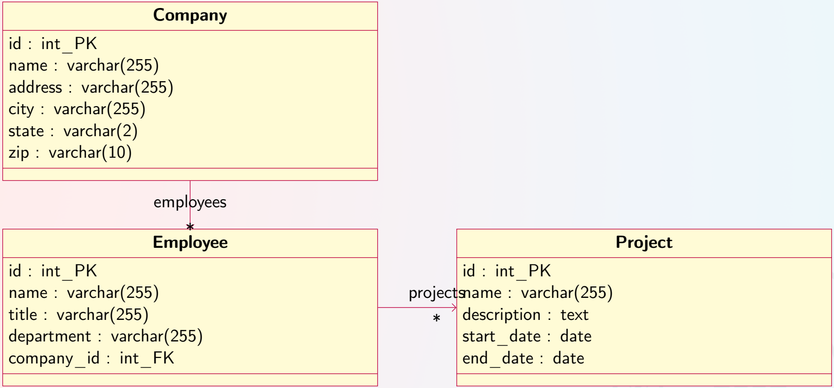
Foreign key（外键）

constraint: value in one relation must appear in another

Foreign Key.

```sql
CREATE TABLE Students (
  StudentID INT PRIMARY KEY,
  FirstName VARCHAR(50),
  LastName VARCHAR(50),
);
CREATE TABLE Courses (
  CourseID INT PRIMARY KEY,
  CourseName VARCHAR(50),
);
CREATE TABLE Enrollments (
  EnrollmentID INT PRIMARY KEY,
  StudentID INT,
  CourseID INT,
  FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
  FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
);
```

Schema diagram



Five Basic Operations of Relational Algebra

1. **Selection（选择）**：selects a subset of rows that satisfy a given condition $\sigma_C(R)$
2. **Projection（投影）**：selects a subset of columns from a table $\Pi_{A_1,A_2,\dots,A_n}(R)$
3. **Union（并）**：combines two tables with the same schema into one table $R \cup S$
4. **Product（笛卡尔积）**：returns the cartesian product of two tables$R \times S$
5. **Join（连接）**：combines two tables by matching rows based on a common attribute $R \bowtie_C S$

These operations can be combined to form more complex queries, and are used in SQL and other database query languages to manipulate and retrieve data.

Student_ID	Name	Age
1001	John	22
1002	Alice	20
1003	Bob	21
1004	Emma	19
1005	David	20

例 1

Suppose we want to select all the students who are 20 years old.

$$\sigma_{\text{Age}=20}(\text{Students})$$

Student_ID	Name	Age
1002	Alice	20
1005	David	20

Department	Building
Biology	Biology
Chemistry	Chemistry
Computer Science	Engineering
English	Main Building
Mathematics	Main Building

例 2

Comparisons between two attributes

$$\sigma_{Building=Department}(Department)$$

which gives us the following table:

Department	Building
Biology	Biology
Chemistry	Chemistry

目录

Student ID	Course ID	Grade
001	CSC101	A
001	MAT105	B
002	CSC101	C
003	ENG201	A
003	CSC201	B
004	MAT105	B

例 3

To select only the columns for Student ID and Course ID:

$$\Pi_{\text{Student ID, Course ID}}(\text{Grade})$$

This will give us the following table:

Student ID	Course ID
001	CSC101
001	MAT105
002	CSC101
003	ENG201
003	CSC201
004	MAT105

A	B	C
1	2	3
4	5	6
7	8	9

例 4

Suppose we want to project only the A and B attributes, and divide the value of B by 2:

$$\Pi_{A,B/2}(R)$$

which results in the following relation:

A	B/2
1	1
4	2.5
7	4

目录

1. Structure of Relational Databases
2. Database Schema
3. Keys
4. Schema diagram
5. The Relational Algebra
   - The Select Operation
   - The project operation
   - Cartesian Product Operation
   - The Join Operation
   - Set Operations
   - The Assignment Operation
   - The Rename Operation
   - Composition and Equivalence
6. Exercise
7. solution

海纳百川 有容乃大

Given two relations $R(A_1, A_2, ..., A_m)$ and $S(B_1, B_2, ..., B_n)$, their Cartesian product $R \times S$ is defined as:

$$R \times S = \{(r, s) \mid r \in R, s \in S\}$$

In other words, $R \times S$ contains **all possible combinations** of tuples from R and S.

ID	Name	Department
101	Alice	CS
102	Bob	EE

Course	Instructor
CS101	Smith
EE201	Johnson

The Cartesian product of the two tables above is:

ID	Name	Department	Course
101	Alice	CS	CS101
101	Alice	CS	EE201
102	Bob	EE	CS101
102	Bob	EE	EE201

海纳百川 有容乃大

Table A:

ID	Name
1	John
2	Mary
3	Bob

Table B:

ID	Salary
1	50000
3	70000
4	60000

To perform a join operation, we need to specify a join condition that determines how the two tables are joined.

例 5

We can perform a join on the "ID" attribute of Tables A and B as follows:

$$A \bowtie_{A.ID=B.ID} B$$

This will result in the following table:

ID	Name	Salary
1	John	50000
3	Bob	70000

Name	Age
John	20
Mary	22

Name	Salary
John	50000
Mary	60000

例 6

Join A and B on Name:

$$A \bowtie_{A.Name=B.Name} B$$

Result without duplication:

A.Name	Age	Salary
John	20	50000
Mary	22	60000

目录

Suppose we have two tables:

ID	Name	Age
1	John	20
2	Mary	22
3	Tom	25

ID	Name	Salary
1	John	50000
2	Mary	60000
4	Bob	70000

$$A_{name} \cup B_{name}$$

Name
John
Mary
Tom
Bob

Result of Set Union Operation on "Name" Attribute

Table A:

Name	Age
John	20
Mary	22

Table B:

Name	Salary
John	50000
Tom	60000

$A = \{John, Mary\}$
$B = \{John, Tom\}$
$A \cap B = \{x | x \in A \text{ and } x \in B\} = \{John\}$

Name
John

Table A:

Name	Age
John	20
Mary	22

Table B:

Name	Salary
John	50000
Tom	60000

$A = \{John, Mary\}$
$B = \{John, Tom\}$
$A - B = \{x | x \in A \text{ and } x \notin B\} = \{Mary\}$

Name
Mary

目录

海纳百川 有容乃大

Assume we have a relation A with attributes x, y, and z, and we want to create a new relation B with attributes x and y where $z > 10$. We can use temporary relation variables to achieve this:

$$
\begin{aligned}
T_1 &\leftarrow \sigma_{z>10}(A) \\
T_2 &\leftarrow \Pi_{x,y}(T_1) \\
B &\leftarrow T_2
\end{aligned}
$$

Finally, we **assign** T_2 to the relation variable B.
Note that we could also combine the selection and projection operators into a single expression:

$$B \leftarrow \Pi_{x,y}(\sigma_{z>10}(A))$$

目录

The **rename** operation is used to rename the attributes of a relation:

$$\rho_{new\_attribute\_name_1/new\_attribute\_name_2/.../new\_attribute\_name_n}(R)$$

where ρ is the rename symbol, $new\_attribute\_name_i$ is the new name for the i^{th} attribute of relation R, and n is the number of attributes in R. For example, suppose we have a relation R with attributes `ID`, `Name`, and `Age`. To rename the `Name` attribute to `FullName`:

$$\rho_{ID/FullName/Age}(R)$$

Note that the rename operation does not modify the original relation R, but creates a new relation with the renamed attributes.

Table A:

Emp_ID	Name	Age
1	Alice	25
2	Bob	30
3	Cindy	28

Table B:

ID	Salary
1	50000
2	60000
3	55000

例 7

If we perform the product operation $A \times B$, we will get a redundant table $(Emp\_ID, Name, Age, ID, Salary)$
To avoid this problem

$$A \times \rho_{ID/NAME/AGE}(B)$$

ID	Name	Age	Salary
1	Alice	25	50000
2	Bob	30	60000
3	Cindy	28	55000

目录

Student:

Student ID	Name
1001	Alice
1002	Bob
1003	Charlie
1004	David

Grade:

Course ID	Student ID	Grade
1	1001	A
1	1002	B
1	1003	C
1	1004	D
2	1001	B
2	1002	C
2	1003	D
2	1004	A

例 8

$$\Pi_{\text{Name}}(\sigma_{\text{Grade} = \text{'A'}}(\text{Grade} \times \text{Student}))$$
$$= \Pi_{\text{Name}}(\sigma_{\text{Grade} = \text{'A'}}(\text{Grade} \bowtie_{\text{Student ID}} \text{Student}))$$

Name
Alice
David

Shopping List:

id	product_id	quantity
1	1	2
1	2	1
2	3	3
3	1	1
3	3	2

Product:

product_id	product_name	price
1	Apple	2.5
2	Orange	1.5
3	Banana	0.8

1. Output the product name and price information for items in the shopping list with a quantity of 2 or more. (**输出购物清单中购买数量大于等于 2 的商品名称和单价信息**)

2.Output the product name and price information for items in the shopping list with a quantity of 2 or more and a price of 1 or more.(**输出购物清单中购买数量大于等于 2 且商品单价大于等于 1 元的商品名称和单价信息**)

商品名字可以重复 An online tool to verify your solution https://dbis-uibk.github.io/relax/landing

Given two relations $R(A, B)$ and $S(B)$, the division operation produces a relation $T(A)$ such that T contains each $a \in A$ such that for every $b \in B$, $(a, b) \in R$.
Formally, the division operation is defined as:

$$T = R \div S = \{t.A \mid \forall s.B \in S : (t.A, s.B) \in R\}$$

product_id	discount_percentage
1	0.10
2	0.15
4	0.05

Discounts Table

3. Use the division operation to find the name of products that are not eligible for any discounts. And show how to write the same query without using division 使用除法运算查找不符合任何折扣条件的产品名。并展示如何在不使用除法的情况下编写相同的查询语句（通过这样做，您将使用其他关系代数操作定义除法运算）

Difference vs. Division in Relational Algebra

- An intuitive property of the division operator of the relational algebra is simply that it is **the inverse of the cartesian product**
- The **difference** operation returns all tuples that are in one relation but not in another.
- The **division** operation, on the other hand, is used to find all tuples in one relation that are related to all tuples in another relation.
- In other words, the difference operation compares tuples between two relations, while the division operation compares tuples within a single relation.

Output the product name and price information for items in the shopping list with a quantity of 2 or more.
We can use the following relational algebra operations to get the desired result:

$$R_1 \leftarrow \sigma_{quantity \geq 2}(shopping\_list)$$
$$R_2 \leftarrow R_1 \bowtie_{R_1.product\_id=product\_info.product\_id} product\_info$$
$$R_3 \leftarrow \Pi_{product\_name,price}(R_2)$$

This gives us the desired output:

product_name	price
Apple	2.5
Banana	0.8
Banana	0.8

Given two relations $R(A,B)$ and $S(B)$, the division of R by S produces a relation $Q(A)$ containing all the values of A for which there is no combination of B that satisfies R.

$$R \div S = \Pi_A(R) - \Pi_A(\Pi_A(R) \times S - R)$$

$\Pi_A(R) \times S$: this contains all possible AB pairs.
R: this contains the actual AB pairs.

Reading

Codd, Edgar F. "A relational model of data for large shared data banks." Communications of the ACM 13.6 (1970): 377-387.
Codd, E. F. "The 1981 ACM Turing Award Lecture." Communications (1982).

Thanks
End of Chapter 2