SWE 207
Database Management Systems

~ Relational Algebra ~

CELAL ÇEKEN,

GÖZDE YOLCU ÖZTEL,

VEYSEL HARUN ŞAHİN

SAKARYA
ÜNİVERSİTESİ

Content

- Formal Query Languages
- Relational Algebra and Relational Algebra Operations
  - Selection
  - Projection
  - Set union
  - Set intersection
  - Set difference
  - Cartesian product
  - Join
  - Division
- Examples

Formal Query Languages

- Two formal query languages are frequently used in the relational model: Relational Algebra and Relational Calculus.
  - These languages are formal. There is no interpreter or compiler.
  - These languages form the basis of SQL (Structured Query Language.

Formal Query Languages

- Relational Calculus: It is not procedural. It is declarative. It allows users to declare what they want. How it should be calculated is not stated.
  - Show the number, name and surname information of students registered in Sakarya.

- Relational Algebra: It is procedural and it is clearly stated which steps the queries consist of.
  - Combine the Students and Provinces table.
  - Select students whose province is Sakarya.
  - Show the number, first and last name of these students.

Relational Algebra

- The basic structure in Relational Database Management Systems is relationship. In the relational model, users think in terms of tables, act on tables, and the results are obtained in the table structure.
- Relational algebra is a formal query language.
- With the help of relational algebra, it is interpreted how the database can be queried.
- Queries are formal, unlike SQL. There is no interpreter or compiler.

Relational Algebra Operations

- Selection: The process of selecting records from a table based on one or more criteria.
    - $\sigma_{selection\ criteria}(Table)$
- Symbols used in selection criteria:
    - $\wedge\ (and), \vee\ (or), \neg(not)$
- Select the student whose student ID is 1512B10010 from the Students table.
    - $\sigma_{studentId='2012B10000'}(Students)$
- Select records that are Male and over 18 years old from the Students table.
    - $\sigma_{gender='M'\ \wedge\ age>18}(Students)$

Relational Algebra Operations

- Projection: Only information for specific attributes is selected and displayed. Repeating lines are shown once. If there is more than one attribute, the attributes are separated by commas.
  - $\pi_{attribute\ list}(Table)$
- Show student ID, name and surname information of all records in the Students table.
  - $\pi_{studentId,name,surname}(Students)$
- Show name and surname information of all records with faculty number 12 from the students table.
  - $\pi_{\ name,surname}(\sigma_{facultyId=12}(Students))$

Relational Algebra Operations

- Set Union: It is represented by taking the set union of two tables. The number of attributes of the tables must be the same. The value fields of the attributes of the tables in the same order must be the same. After the cluster merge operation, the duplicate rows are returned once.
  - $Table1 \cup Table2$
- Combine and show the studentId field of records in ComputerCommunity and ElectronicCommunity tables.
  - $\pi_{studentId}(ComputerCommunity) \cup \pi_{studentId}(\text{ElectronicCommunity})$

Relational Algebra Operations

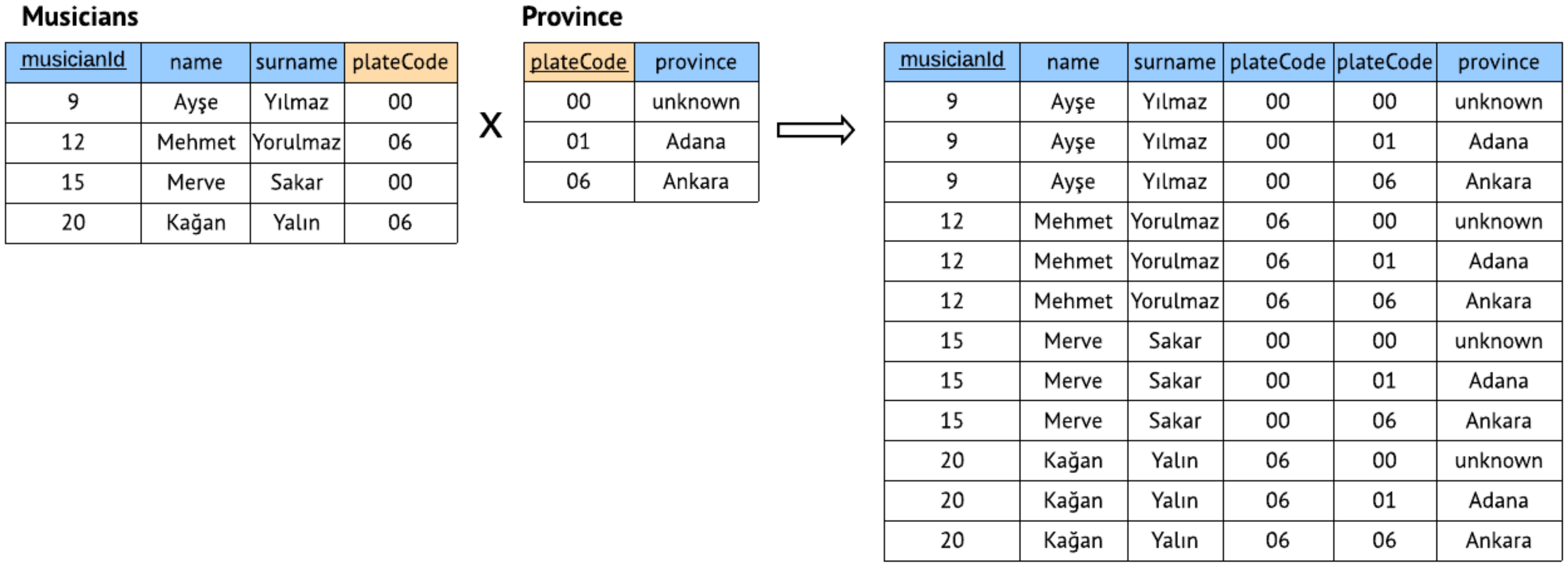
- Set Intersection: It is represented by taking the set intersection of two tables.
- The number of attributes of the tables must be the same.
- The value fields of the attributes of the tables in the same order must be the same.
  - $Table1 \cap Table2$
- Select and show the information in the fields of the records with the same information for the StudentID field in the ComputerCommunity and ElectronicCommunity tables.
  - $\pi_{studentId}(ComputerCommunity) \cap \pi_{studentId}(\text{ElectronicCommunity})$

Relational Algebra Operations

- Set Difference: It is shown by taking the set difference of two tables. The number of attributes of the tables must be the same. The value fields of the attributes of the tables in the same order must be the same.
  - $Table1 - Table2$
- Show the information for students who are in the ComputerCommunity table but not in the ElectronicCommunity table.
  - $\pi_{studentId}(ComputerCommunity) - \pi_{studentId}(\text{ElectronicCommunity})$

Relational Algebra Operations

- Cartesian Product: Cartesian multiplication of two tables is performed.
- It applies to two different tables.
  - $Table1\ x\ Table2$



Musicians

musicianId	name	surname	plateCode
9	Ayşe	Yılmaz	00
12	Mehmet	Yorulmaz	06
15	Merve	Sakar	00
20	Kağan	Yalın	06

X

Province

plateCode	province
00	unknown
01	Adana
06	Ankara

⟹

musicianId	name	surname	plateCode	plateCode	province
9	Ayşe	Yılmaz	00	00	unknown
9	Ayşe	Yılmaz	00	01	Adana
9	Ayşe	Yılmaz	00	06	Ankara
12	Mehmet	Yorulmaz	06	00	unknown
12	Mehmet	Yorulmaz	06	01	Adana
12	Mehmet	Yorulmaz	06	06	Ankara
15	Merve	Sakar	00	00	unknown
15	Merve	Sakar	00	01	Adana
15	Merve	Sakar	00	06	Ankara
20	Kağan	Yalın	06	00	unknown
20	Kağan	Yalın	06	01	Adana
20	Kağan	Yalın	06	06	Ankara

Relational Algebra Operations

- Natural Join/Inner Join: The subset of the cartesian product of the two tables is obtained. Selection is applied to the Cartesian product result. Records with the same information for the same attribute are selected and displayed.
  - $Table1 \bowtie Table2$

Musicians

musicianId	name	surname	plateCode
9	Ayşe	Yılmaz	00
12	Mehmet	Yorulmaz	06
15	Merve	Sakar	00
18	Hale	Çınar	54
20	Kağan	Yalın	06

⋈

Province

plateCode	province
00	unknown
01	Adana
06	Ankara
34	İstanbul

⇒

musicianId	name	surname	plateCode	plateCode	province
9	Ayşe	Yılmaz	00	00	unknown
12	Mehmet	Yorulmaz	06	06	Ankara
15	Merve	Sakar	00	00	unknown
20	Kaan	Yalın	06	06	Ankara

Relational Algebra Operations

- <u>Left Outer Join:</u>
  - It applies to two tables.
  - All records in the table on the left are retrieved.
  - Records with the same information of the same attribute are selected from the table on the right, and their information is added to the information from the table on the left.
  - If there are no matching records in the table on the right, they are left blank (NULL).

  - $Table1 ⟕ Table2$

Relational Algebra Operations

- Left Outer Join:
  - $Table1$ ⟕ $Table2$

Musicians

musicianId	name	surname	plateCode
9	Ayşe	Yılmaz	33
12	Mehmet	Yorulmaz	06
15	Merve	Sakar	00
20	Kağan	Yalın	06
22	Cenk	Dur	07

⟕

Province

plateCode	province
00	unknown
01	Adana
06	Ankara

⇒

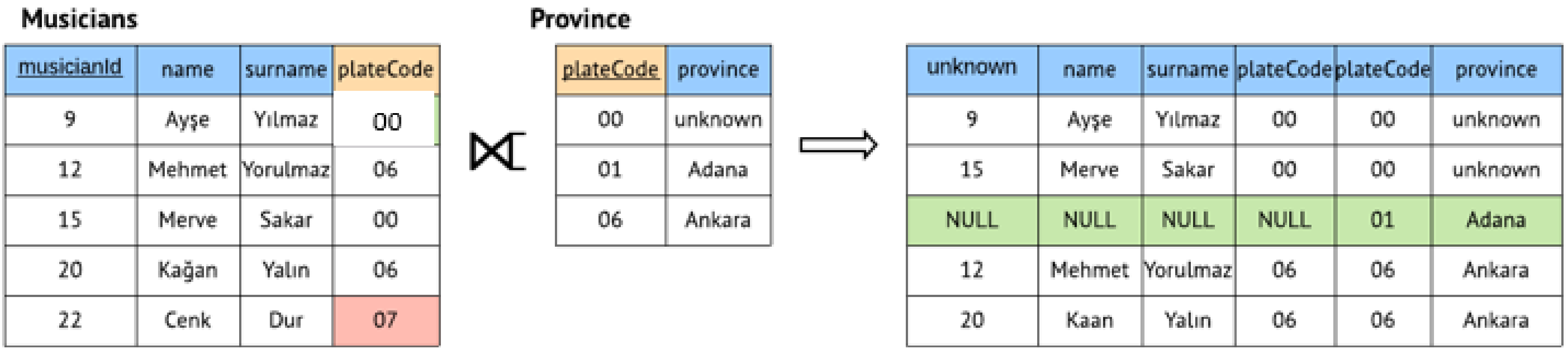
musicianId	name	surname	plateCode	plateCode	province
9	Ayşe	Yılmaz	33	NULL	NULL
12	Mehmet	Yorulmaz	06	06	Ankara
15	Merve	Sakar	00	00	unknown
20	Kaan	Yalın	06	06	Ankara
22	Cenk	Dur	07	NULL	NULL

Relational Algebra Operations

- Right Outer Join:
  - It applies to two tables.
  - All records in the table on the right are retrieved.
  - Records with the same information of the same attribute are selected from the table on the leftt, and their information is added to the information from the table on the right.
  - If there are no matching records in the table on the left, they are left blank (NULL).
  - $Table1 ⟖ Table2$

Relational Algebra Operations

- <u>Right Outer Join:</u>
  - $Table1$ ⟖ $Table2$



Musicians

musicianId	name	surname	plateCode
9	Ayşe	Yılmaz	00
12	Mehmet	Yorulmaz	06
15	Merve	Sakar	00
20	Kağan	Yalın	06
22	Cenk	Dur	07

⟖

Province

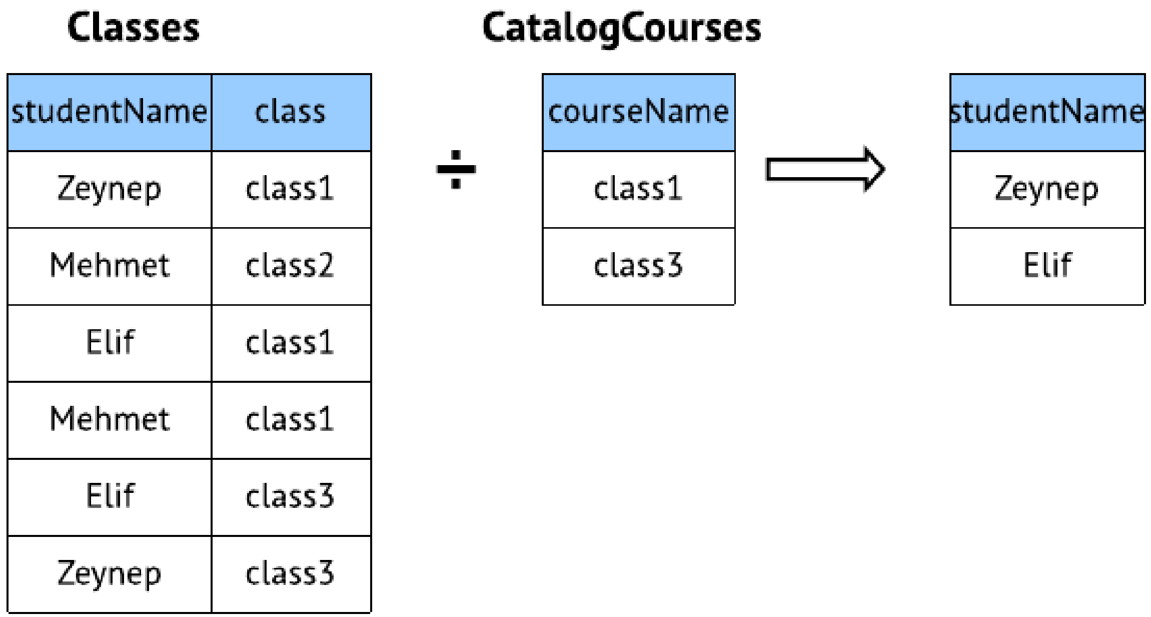
plateCode	province
00	unknown
01	Adana
06	Ankara

⟹

unknown	name	surname	plateCode	plateCode	province
9	Ayşe	Yılmaz	00	00	unknown
15	Merve	Sakar	00	00	unknown
NULL	NULL	NULL	NULL	01	Adana
12	Mehmet	Yorulmaz	06	06	Ankara
20	Kaan	Yalın	06	06	Ankara

Relational Algebra Operations

- Divison: It applies to two tables. The two tables must have a common attribute.
  - $Table1 \div Table2$



Classes

studentName	class
Zeynep	class1
Mehmet	class2
Elif	class1
Mehmet	class1
Elif	class3
Zeynep	class3

÷

CatalogCourses

courseName
class1
class3

⇒

studentName
Zeynep
Elif

Examples

- The relational schema (sub schema) of a University Information System database is as follows. Make the formal expressions of the desired queries with relational algebra.
  - Department (departmentId: int, name: varchar(20))
  - Student (studentId: char(10), name: varchar(20), surname: varchar(20), birthDate: date, birthPlace: varchar(20), plateno: varchar(16), gender: char(1), departmentId: int)
  - Classes (classId: char(6), courseId: char(5), term: varchar(10), registerId: char(7), departmentId: int)
  - Enroll (enrollId: int, classId: char(6), studentId: char(10), midtermExam: numeric, finalExam: numeric, avg: numeric)
  - facultyMember (registerId: char(7), name: varhcar(20), surname: varchar(20), birthDate: date, birthPlace: char(16), title: varchar(20))

Examples

- Write the relational algebra expression that lists the student numbers, first and last names of all students.
  - $\pi_{studentId,name,surname}(Student)$
- Write the relational algebra expression that lists the names and surnames of the students born in Sakarya.
  - $\pi_{\ name,surname}(\sigma_{birthPlace='Sakarya'}(Student))$

Examples

- Write the relational algebra expression that lists the registration numbers, names and surnames of the faculty members who have courses in the 2000-2001 academic year.

  - $\pi_{registrationId,name,surname}(FacultyMember \ \bowtie (\sigma_{term='2000-2001'}(Classes))$

Examples

- Write the relational algebra expression that lists the registration numbers, names and surnames of the faculty members who did not have any courses in the 2000-2001 academic year.

  - $\pi_{registrationId,name,surname}(FacultyMember)$ - $\pi_{registrationId,name,surname}(FacultyMember \ \bowtie (\sigma_{term='2000-2001'}(Classes))$

Examples

- Write the relational algebra expression that lists the names and surnames of the faculty members teaching in the Computer Engineering department.

  - $\pi_{name,surname}(FacultyMember \bowtie Classes \bowtie (\sigma_{name='CS'}(Department))$