Databases Design. Introduction to SQL

LECTURE 7

Relational algebra

SQL Structure

$$\begin{tikzcd}
 & & \text{SQL} \arrow[dll] \arrow[ddl] \arrow[ddr] \arrow[drr] & & \\
\text{DDL} & & & & \text{DCL} \\
 & \text{DML} & & \text{TCL} &
\end{tikzcd}$$

- DDL (Data Definition Language)
- DML (Data Manipulation Language)
- TCL (Transaction Control Language)
- DCL (Data Control Language)

Last lecture

A **DML** is a language which enables to access and manipulate data.

DML statements:

- INSERT
- UPDATE
- DELETE
- SELECT

Querying Data From Tables

- **Query operations** facilitate data retrieval from one or more tables.

- The result of any query is a **table**.
- The result can be further manipulated by other query operations.

Querying Data From Tables

- SQL allows to query data using SELECT statement.

Syntax:

```
SELECT attribute(s)
FROM table(s)
[WHERE selection condition(s)] ;
```

Relational algebra

- **Relational algebra**, first described by E.F. Codd, is a family of algebras with a well-founded semantics used for modelling the data stored in relational databases, and defining queries on it.
- **Relational algebra** is a theoretical language with operations that work on one or more relations to define another relation without changing the original relation(s).
- Once the data is normalized in sets of data (entities), the operations of the relational algebra can be performed.

Relational algebra

- Similar to normal algebra, except we use relations as values instead of numbers, and the operations and operators are different.
- The main application of relational algebra is providing a **theoretical foundation** for relational databases.
- Not used as a query language in actual DBMSs (SQL instead).
- We need to know about relational algebra to understand query execution in a relational DBMS.

Operations

Operations of Relational algebra:

- projection
- selection
- union
- difference (set difference)
- intersection
- join
- cartesian product

Projection

Projection, referred to as Π(pi)

- Selects a set of attributes from a table
- The attributes are subscripts to Π and the table is in parentheses

Π_{stud_id} (Students)

- Projection is represented in a SQL SELECT statement's attribute list. The above projection is synonymous to the following SQL query:

```
SELECT stud_id
FROM Students;
```

Selection

Selection, referred to as σ (sigma)

- Selects a set of rows from a table that satisfy a selection condition
- The selection condition is the subscript to σ and the table is in parenthesis

$$\sigma_{stud\_id=01}\ (Students)$$

Selection

- In SQL, selection is represented in the WHERE clause of a select statement.
- Translate $\sigma_{stud\_id=01}$ (Students) to SQL:

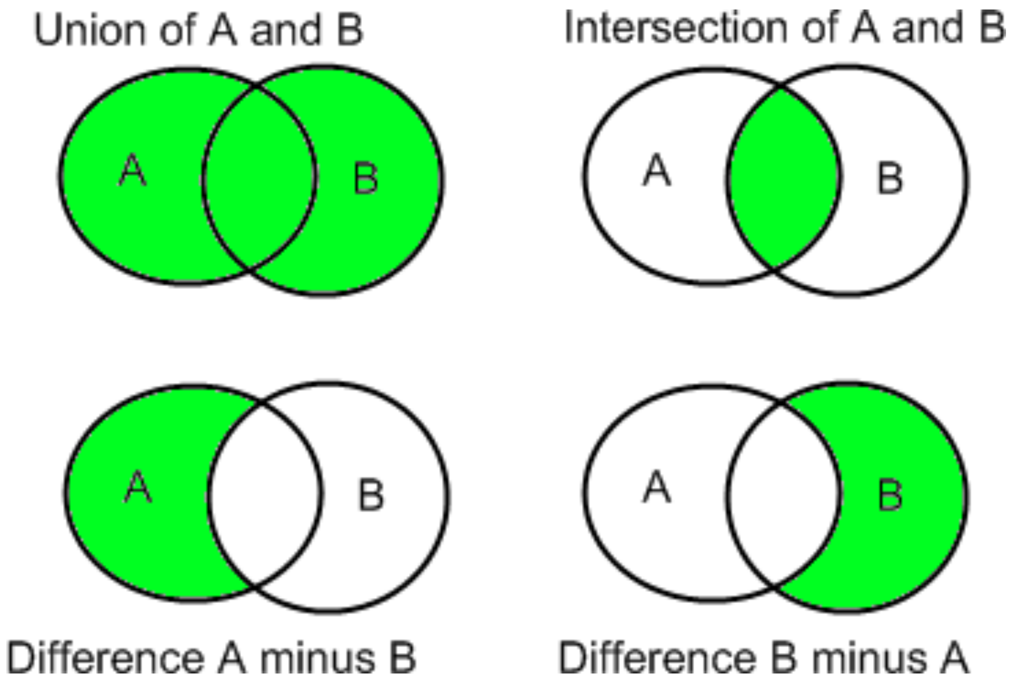
```
SELECT *
FROM Students
WHERE stud_id=01;
```

- What does SELECT * mean?

It means that we are selecting all data – all attributes - from a table.

Union, Difference, Intersection

- **Union** (R1 U R2) is the relation containing all tuples that appear in R1, R2, or both.
- **Set difference** (R1 - R2) is the relation containing all tuples of R1 that do not appear in R2.
- **Intersection** (R1 ∩R2) is the relation containing all tuples that appear only in both R1 and R2.



Union-compatible

Two tables must be **union-compatible** for the operations to work:

- Tables need to have same number of attributes
- The domain of each attribute must also be the same.

Union-compatible: example

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

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

Support in SQL

- For **Union** SQL supports the UNION operator.
- For **Difference** (or **Set Difference**) SQL supports the EXCEPT operator.
- For **Intersection** SQL supports the INTERSECT operator.

Combining Queries

The results of two queries can be combined using the set operations **union**, **intersection**, and **difference**.

The syntax is

```
query1 UNION [ALL] query2
query1 INTERSECT [ALL] query2
query1 EXCEPT [ALL] query2
```

query1 and ***query2*** are queries that can use any of the features discussed up to this point.

Combining Queries

Set operations can also be nested and chained, for example

```
query1 UNION query2 UNION query3
```

which is executed as:

```
(query1 UNION query2) UNION query3
```

In order to calculate the union, intersection, or difference of two queries, the two queries must be "**union compatible**", which means that they return the same number of columns and the corresponding columns have compatible data types.

Union / UNION

- The UNION operation on relation A UNION relation B designated as **A** ∪ **B**, includes all tuples that are in A or in B, eliminating duplicate tuples.
- To include duplicates, use the UNION ALL operator.

SQL Syntax:

```
SELECT * From A
UNION
SELECT * From B
```

UNION

```
SELECT * From R
UNION
SELECT * From S
```

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

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

Table R U S		
A	B	C
1	2	3
4	5	6
7	8	9
1	3	2
8	7	9

UNION ALL

SELECT * From R
UNION ALL
SELECT * From S

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

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

Table R U S		
A	B	C
1	2	3
4	5	6
7	8	9
1	3	2
4	5	6
8	7	9

Set Difference / EXCEPT

- The DIFFERENCE operation includes tuples from one relation that are not in another relation.
- Let the Relations be A and B, the operation A EXCEPT B is denoted by **A – B**, that results in tuples that are A and not in B.

  SQL Syntax:

  ```
  SELECT * FROM A
  EXCEPT
  SELECT * FROM B
  ```

EXCEPT

```
SELECT * FROM R
EXCEPT
SELECT * FROM S
```

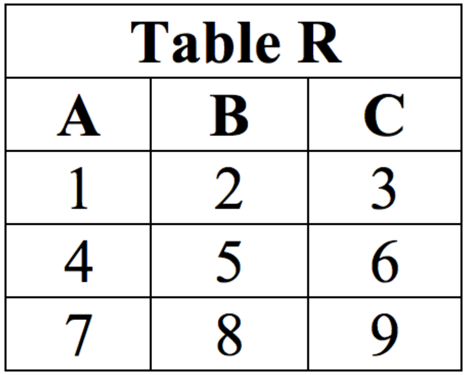


Table R		
A	B	C
1	2	3
4	5	6
7	8	9

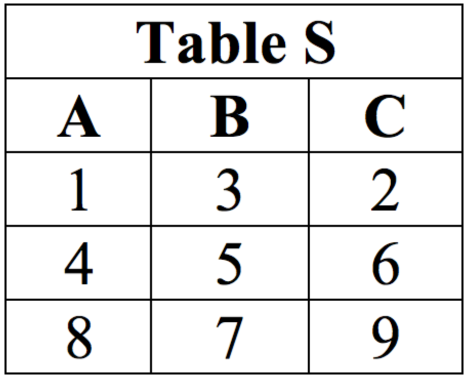


Table S		
A	B	C
1	3	2
4	5	6
8	7	9

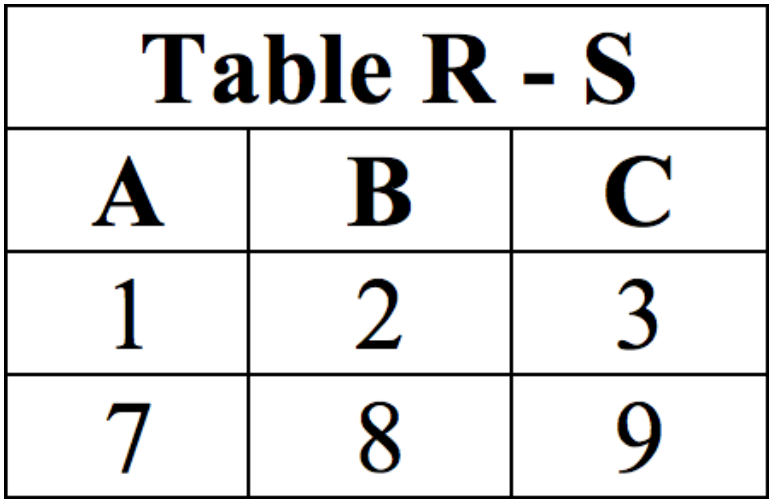


Table R - S		
A	B	C
1	2	3
7	8	9

EXCEPT

```
SELECT * FROM S
EXCEPT
SELECT * FROM R
```

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

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

Table S - R		
A	B	C
1	3	2
8	7	9

Intersection / INTERSECT

- The INTERSECTION operation on a relation A INTERSECT relation B, designated by **A∩B**, includes tuples that are only in A and B.
- In other words only tuples belonging to A and B, or shared by both A and B are included in the result.

SQL Syntax:

```
SELECT * FROM A
INTERSECT
SELECT * FROM B
```

INTERSECT

```
SELECT * FROM R
INTERSECT
SELECT * FROM S
```

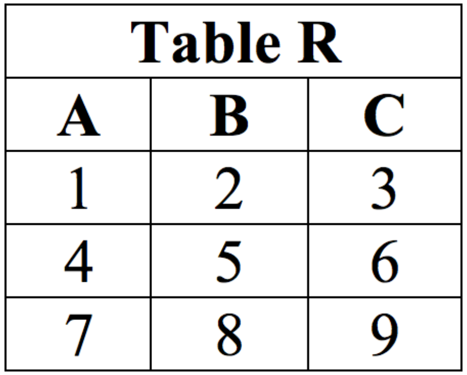


Table R		
A	**B**	**C**
1	2	3
4	5	6
7	8	9

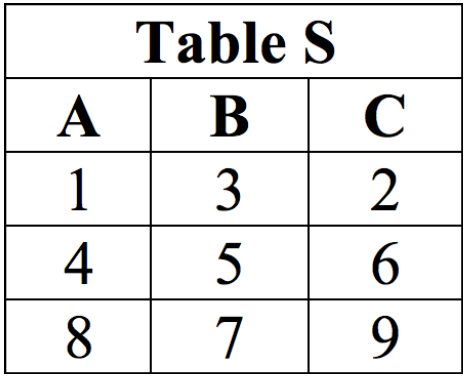


Table S		
A	**B**	**C**
1	3	2
4	5	6
8	7	9

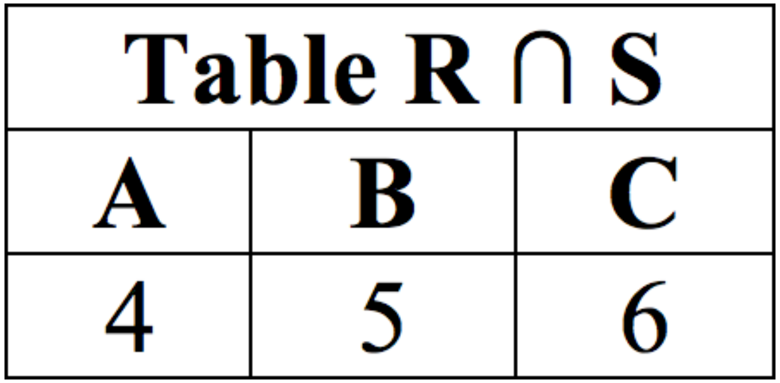


Table R ∩ S		
A	**B**	**C**
4	5	6

Books

- **Connolly, Thomas M. Database Systems**: A Practical Approach to Design, Implementation, and Management / Thomas M. Connolly, Carolyn E. Begg.- United States of America: Pearson Education

- **Garcia-Molina, H. Database system**: The Complete Book / Hector Garcia-Molina.- United States of America: Pearson Prentice Hall

- **Sharma, N. Database Fundamentals**: A book for the community by the community / Neeraj Sharma, Liviu Perniu.- Canada

- www.postgresql.org/docs/manuals/