**Relational algebra**

**Objectives:**

* Extract data from various relations, joins
* Describe and use basic operations
* Identify syntax for various operations
* Translate relational algebra expressions to SQL statements (vice versa)

**JOINS**

Used to ***combine*** data from ***two different tables*** into one temporary table.

Theta or inner joins will only combine data when a given condition is met.

**INNER JOINS** select data that have the ***same*** records in ***both*** tables:

*SELECT column1*

*FROM table1*

*INNER JOIN table2*

*ON table1.column1 = table2.column2*

**LEFT JOIN** only selects data from the left table that have matched entries in the right table:

*SELECT column1*

*FROM table1*

*LEFT JOIN table2*

*ON table1.column1 = table2.column2*

**RIGHT JOIN** simply *LEFT JOIN* but vice versa.

*SELECT column1*

*FROM table1*

*RIGHT JOIN table2*

*ON table1.column1 = table2.column2*

**FULL OUTER JOIN** selects data whenever there is a match in the left or right table.

*SELECT column1*

*FROM table1*

*FULL OUTER JOIN table2*

*ON table1.column1 = table2.column2*

**SELF JOIN** is used whenever you want to join a table to itself.

*SELECT column1*

*FROM table1, table2*

*WHERE condition*

**NATURAL JOIN**

The operator looks for ***common columns*** in the two tables, of which it ***combines*** the values of those common columns that are found in both tables ***into one table***.

**Relational algebra**

Relation algebra contains a set of theoretical operations used to manipulate data in relational databases.

***5*** Basic Operations:

* ***UNION – U***
* ***SET DIFFERENCE – ‘-‘***
* ***SELECTION - ?***
* ***CARTESIAN PRODUCT - x***
* ***PROJECTION - π***

**UNION**

Combines the result of ***two select statements*** of which the two tables should have:

* same number of fields
* corresponding fields have the same data type
* duplicates rows are automatically eliminated.

*SELECT column1, column2, ... FROM table1*

*UNION*

*SELECT column1, column2, ... FROM table2*

**SET DIFFERENCE/EXCEPT**

Returns the rows, which are contained in table1 ***but not*** in table2

*SELECT column1, column2, ... FROM table1*

*EXCEPT*

*SELECT column1, column2, … FROM table2;*

**CROSS JOIN (CARTESIAN-PRODUCT)**

Combines ***each row*** from table1 with each row with table2, resulting in a new table ***containing every possible combination*** of the rows between the tables.

*SELECT \* FROM table1*

*CROSS JOIN*

*table2*

**PROJECTION**

Similar to *SELECT* but it only returns data from ***specified*** columns

*SELECT column1, column2, ... FROM table1*

**INTERSECTION**

Takes two tables that are *union-compatible* and only returns the rows which ***occur*** in both tables.

*SELECT column1, column2, ... FROM table1*

*INTERSECT*

*SELECT column1, column2, … FROM table2*

**Select, Project and Rename operator**

1. ***Selection ()*** – ***filters*** and ***select*** ***rows*** out of a table based on a **condition.**
2. ***Project () – selects*** only ***specified*** columns out of a row in a table.
3. ***Rename () –*** used ***rename attributes or* relations** to something else

**Select operator**

R - Relation

Syntax:

Examples:

”)Team

// This will select all the rows in the **Team** table where the **City** column has a value of “Rundu”

SELECT \* FROM Team WHERE City=”Rundu”

// SELECT \* FROM HACKERS WHERE HackerSkill IN (‘Networking’,’Programming’) AND hackerID >= 9001;

SELECT \* FROM Minster WHERE Minster\_martial\_status=’M’ OR salary > 5000;

**Project operator**

Syntax:

Examples:

//SELECT hackerID, hackerName, hackerSkill FROM Hackers;

**Rename operator**

**Rename a relation**

Syntax:

S – new name of relation

R – relation name to modify

Example:

//Changes the name of the relation Hackers to EliteHackers

**Rename a attribute**

Syntax:

OR

Examples:

// Changes the 1st and 2nd attribute to ID and Port in the Hackers table

**Rename a relation and its attributes**

Syntax:

S(newAttr1, newAttrn)R

Example:

// Renames the Hackers table to hacker, and its 1st and 2nd attributes to ID and Skill