SQL (Structured Query Language) is a powerful language for managing and manipulating relational databases. Relational algebra, on the other hand, is a theoretical language used to express queries on relational databases. SQL statements can be translated into relational algebra expressions. Here are some common SQL statements and their corresponding relational algebra expressions:

### 1. \*\*Selection (σ) - SQL WHERE Clause:\*\*

\*\*SQL:\*\*

```sql

SELECT column1, column2

FROM table

WHERE condition;

```

\*\*Relational Algebra:\*\*

\[ \sigma\_{\text{condition}}(\text{table}) \]

### 2. \*\*Projection (π):\*\*

\*\*SQL:\*\*

```sql

SELECT column1, column2

FROM table;

```

\*\*Relational Algebra:\*\*

\[ \pi\_{\text{column1, column2}}(\text{table}) \]

### 3. \*\*Union (∪):\*\*

\*\*SQL:\*\*

```sql

SELECT column1, column2

FROM table1

UNION

SELECT column1, column2

FROM table2;

```

\*\*Relational Algebra:\*\*

\[ \text{table1} \cup \text{table2} \]

### 4. \*\*Intersection (∩):\*\*

\*\*SQL:\*\*

```sql

SELECT column1, column2

FROM table1

INTERSECT

SELECT column1, column2

FROM table2;

```

\*\*Relational Algebra:\*\*

\[ \text{table1} \cap \text{table2} \]

### 5. \*\*Difference (-):\*\*

\*\*SQL:\*\*

```sql

SELECT column1, column2

FROM table1

EXCEPT

SELECT column1, column2

FROM table2;

```

\*\*Relational Algebra:\*\*

\[ \text{table1} - \text{table2} \]

### 6. \*\*Cartesian Product (×):\*\*

\*\*SQL:\*\*

```sql

SELECT \*

FROM table1, table2;

```

\*\*Relational Algebra:\*\*

\[ \text{table1} \times \text{table2} \]

### 7. \*\*Join (⨝):\*\*

\*\*SQL:\*\*

```sql

SELECT column1, column2

FROM table1

JOIN table2 ON table1.key = table2.key;

```

\*\*Relational Algebra:\*\*

\[ \text{table1} \bowtie\_{\text{table1.key = table2.key}} \text{table2} \]

### 8. \*\*Group By (γ):\*\*

\*\*SQL:\*\*

```sql

SELECT column1, COUNT(column2)

FROM table

GROUP BY column1;

```

\*\*Relational Algebra:\*\*

\[ \gamma\_{\text{column1, COUNT(column2)}}(\text{table}) \]

These are simplified examples, and in a real-world scenario, SQL queries and relational algebra expressions can become more complex. The translation from SQL to relational algebra may involve additional operators and transformations based on the specifics of the query.

Let's consider a more complex example involving multiple tables and operations. Suppose we have two tables: "Employees" and "Departments," and we want to find the average salary for each department with more than five employees. Here's how we can express this in both SQL and relational algebra:

### Tables:

\*\*Employees:\*\*

```

| EmpID | Name | Salary | DeptID |

|-------|--------|--------|--------|

| 1 | Alice | 60000 | 101 |

| 2 | Bob | 70000 | 102 |

| 3 | Charlie| 55000 | 101 |

| 4 | David | 80000 | 103 |

| 5 | Eve | 75000 | 102 |

| 6 | Frank | 60000 | 101 |

| 7 | Grace | 72000 | 103 |

```

\*\*Departments:\*\*

```

| DeptID | DeptName |

|--------|-------------|

| 101 | HR |

| 102 | IT |

| 103 | Finance |

```

### SQL:

```sql

SELECT Departments.DeptID, DeptName, AVG(Salary) AS AvgSalary

FROM Departments

JOIN Employees ON Departments.DeptID = Employees.DeptID

GROUP BY Departments.DeptID, DeptName

HAVING COUNT(EmpID) > 5;

```

### Relational Algebra:

\[ \pi\_{\text{DeptID, DeptName, AvgSalary}} \left( \gamma\_{\text{DeptID, DeptName, AVG(Salary) AS AvgSalary}} \left( \sigma\_{\text{COUNT(EmpID) > 5}} \left( \rho\_{\text{Employees(DeptID, EmpID, Name, Salary)}} \right) \bowtie\_{\text{DeptID = Departments.DeptID}} \rho\_{\text{Departments(DeptID, DeptName)}} \right) \right) \]

In this example:

- \*\*σ (Selection):\*\* Filters the employees to include only those in departments with more than five employees.

- \*\*ρ (Rename):\*\* Renames the Employees table for better readability.

- \*\*⨝ (Join):\*\* Joins the renamed Employees table with the Departments table based on the DeptID.

- \*\*γ (Group By):\*\* Groups the result by DeptID and calculates the average salary for each department.

- \*\*σ (Selection):\*\* Filters the result to include only departments with more than five employees.

- \*\*π (Projection):\*\* Selects the final columns for the output.

This example demonstrates the use of various relational algebra operations to achieve a complex query involving multiple tables and aggregation.