Each of the following are depictions of the **same** relation, the **dept** relation, whose relation structure form can be given as:

**dept(dept_name, DEPT_NUM, dept_loc)**

| dept_name | dept_num | dept_loc |
|---|---|---|
| Accounting | 10 | New York |
| Research | 20 | Dallas |
| Sales | 30 | Chicago |
| Operations | 40 | Boston |

| dept_num | dept_name | dept_loc |
|---|---|---|
| 40 | Operations | Boston |
| 30 | Sales | Chicago |
| 20 | Research | Dallas |
| 10 | Accounting | New York |

# Relational selection example

For example, say that you have a **Student** relation as follows:

**STUDENT table:**

| Stud_ID | Stud_LName | Stud_Major | Stud_Grade_Level | Stud_Age |
|---|---|---|---|---|
| 123 | Jones | History | JR | 21 |
| 158 | Parks | Math | GR | 26 |
| 105 | Anderson | Management | SR | 27 |
| 271 | Smith | History | JR | 19 |

Then the result of the **selection** operation on the **Student** table of rows in which **Stud_Age < 25** would be:

**STUDENT WHERE Stud_Age < 25**

| Stud_ID | Stud_LName | Stud_Major | Stud_Grade_Level | Stud_Age |
|---|---|---|---|---|
| 123 | Jones | History | JR | 21 |
| 271 | Smith | History | JR | 19 |

# Relational projection example

For example, say that you have a **Student** relation as follows:

**STUDENT table:**

| Stud_ID | Stud_LName | Stud_Major | Stud_Grade_Level | Stud_Age |
|---|---|---|---|---|
| 123 | Jones | History | JR | 21 |
| 158 | Parks | Math | GR | 26 |
| 105 | Anderson | Management | SR | 27 |
| 271 | Smith | History | JR | 19 |

Then the result of the **projection** operation of the **Student_Major** and **Stud_Grade_Level** attributes of the **Student** table would be:

| Stud_Major | Stud_Grade_Level |
|---|---|
| History | JR |
| Math | GR |
| Management | SR |

# Relational Cartesian product example

For example, say that you have a **Price** relation as follows:

| Prod_Code | Price |
|---|---|
| AA | 5.99 |
| BB | 22.75 |

And, say that you have a **Location** table as follows:

| Store | Aisle | Shelf |
|---|---|---|
| 23 | W | 5 |
| 24 | K | 9 |
| 25 | Z | 6 |

Then the **Cartesian product** of Price and Location would be:

| Prod_Code | Price | Store | Aisle | Shelf |
|---|---|---|---|---|
| AA | 5.99 | 23 | W | 5 |
| AA | 5.99 | 24 | K | 9 |
| AA | 5.99 | 25 | Z | 6 |
| BB | 22.75 | 23 | W | 5 |
| BB | 22.75 | 24 | K | 9 |
| BB | 22.75 | 25 | Z | 6 |

# Relational equi-join and natural join examples

For example, say that you have **Student** and **Enrollment** tables as follows:

**STUDENT table:**

| Stud_ID | Stud_LName | Stud_Major | Stud_Grade_Level | Stud_Age |
|---|---|---|---|---|
| 123 | Jones | History | JR | 21 |
| 158 | Parks | Math | GR | 26 |
| 105 | Anderson | Management | SR | 27 |
| 271 | Smith | History | JR | 19 |

**ENROLLMENT table:**

| Stud_ID | Class_Name | Position_Num |
|---|---|---|
| 123 | H350 | 1 |
| 105 | BA490 | 3 |
| 123 | BA490 | 7 |

Say that you wish to compute the **equi-join** and **natural join** of these tables based on the join condition (**Student.stud_id = Enrollment.stud_id)**.

**First**, compute the **Cartesian product** of these two tables:

| Student.Stud_ID | Stud_LName | Stud_Major | Stud_Grade_Level | Stud_Age | Enrollment.Stud_ID | Class_Name | Position_Num |
|---|---|---|---|---|---|---|---|
| 123 | Jones | History | JR | 21 | 123 | H350 | 1 |
| 123 | Jones | History | JR | 21 | 105 | BA490 | 3 |
| 123 | Jones | History | JR | 21 | 123 | BA490 | 7 |
| 158 | Parks | Math | GR | 26 | 123 | H350 | 1 |
| 158 | Parks | Math | GR | 26 | 105 | BA490 | 3 |
| 158 | Parks | Math | GR | 26 | 123 | BA490 | 7 |
| 105 | Anderson | Management | SR | 27 | 123 | H350 | 1 |
| 105 | Anderson | Management | SR | 27 | 105 | BA490 | 3 |
| 105 | Anderson | Management | SR | 27 | 123 | BA490 | 7 |
| 271 | Smith | History | JR | 19 | 123 | H350 | 1 |
| 271 | Smith | History | JR | 19 | 105 | BA490 | 3 |
| 271 | Smith | History | JR | 19 | 123 | BA490 | 7 |

**Second**, perform a selection on this result of only those rows for which (**Student.stud_id = Enrollment.stud_id)**:

| Student.Stud_ID | Stud_LName | Stud_Major | Stud_Grade_Level | Stud_Age | Enrollment.Stud_ID | Class_Name | Position_Num |
|---|---|---|---|---|---|---|---|
| 123 | Jones | History | JR | 21 | 123 | H350 | 1 |
| 123 | Jones | History | JR | 21 | 123 | BA490 | 7 |
| 105 | Anderson | Management | SR | 27 | 105 | BA490 | 3 |

This is the **equi-join** of these two tables on the join condition **(Student.stud_id = Enrollment.stud_id)**.

Then, the **natural join** of these two tables on this join condition would include the **third** step of now projecting all of the columns in this result except for one of the "duplicate-contents" columns (and it doesn't matter which one of the two is omitted).

So, for example, the resulting **natural join** in this case could be:

| Stud_ID | Stud_ LName | Stud_ Major | Stud_ Grade_ Level | Stud_ Age | Class_ Name | Position_ Num |
|---|---|---|---|---|---|---|
| 123 | Jones | History | JR | 21 | H350 | 1 |
| 123 | Jones | History | JR | 21 | BA490 | 7 |
| 105 | Anderson | Management | SR | 27 | BA490 | 3 |