Expressing Conditional and Boolean Queries in Relational Algebra

Dirk Van Gucht[1]

[1] Indiana University

Outline

Objectives:

- Expressing *conditional*, i.e., "`if-then-else`" queries in Relational Algebra and SQL with RA operations.
- Expressing *boolean* queries in Relational Algebra and SQL with RA operations.
- Translating boolean SQL queries into boolean RA expressions.

The "`if-then-else`" conditional query

Develop an RA expression for the "`if-then-else`" query

$$\texttt{if} \quad \mathcal{C}(F) \quad \texttt{then} \quad E_1$$
$$\texttt{else} \quad E_2$$

F	RA expression with schema **A**
E_1	RA expression with schema **B**
E_2	RA expression with schema **B**
$\mathcal{C}(F)$	a boolean set/relation condition $\mathcal{C}$ on F

Typical cases for $\mathcal{C}(F)$:

$$F \neq \emptyset$$
$$F = \emptyset$$
$$|F| \quad \theta \quad \text{k} \quad \text{with } \theta \text{ one of } =, \neq, <, \leq, >, \geq$$

Semantics of the “`if-then-else`” query

The semantics of the “`if-then-else`" query

$$\texttt{if} \quad \mathcal{C}(F) \quad \begin{array}{ll} \texttt{then} & E_1 \\ \texttt{else} & E_2 \end{array}$$

If $\mathcal{C}(F)$ is true then the “`if-then-else`" query returns the value of the expression E_1.

If $\mathcal{C}(F)$ is false then the “`if-then-else`" query returns the value of the expression E_2.

Expressing "`if-then-else`" query in Relational Algebra

We begin with a special case of the "`if-then-else`" query, i.e.[1]

$$\texttt{if} \quad F \neq \emptyset \quad \texttt{then} \quad E_1$$
$$\texttt{else} \quad E_2$$

This query can be expressed in RA with the expression

$$\pi_{\mathbf{B}}(E_1 \times F) \cup (E_2 - \pi_{\mathbf{B}}(E_2 \times F)).$$

[1] Recall that E_1 and E_2 both have schema **B**.

Expressing "`if-then-else`" query in Relational Algebra

$$\pi_{\mathbf{B}}(E_1 \times F) \cup (E_2 - \pi_{\mathbf{B}}(E_2 \times F))$$

- If $F \neq \emptyset$, then $\pi_{\mathbf{B}}(E_1 \times F) = E_1$ and $\pi_{\mathbf{B}}(E_2 \times F) = E_2$. Therefore,

  $$\pi_{\mathbf{B}}(E_1 \times F) \cup (E_2 - \pi_{\mathbf{B}}(E_2 \times F) = E_1 \cup (E_2 - E_2) = E_1 - \emptyset = E_1 .$$

- If $F = \emptyset$, then $\pi_{\mathbf{B}}(E_1 \times F) = \emptyset$ and $\pi_{\mathbf{B}}(E_2 \times F) = \emptyset$. Therefore,

  $$\pi_{\mathbf{B}}(E_1 \times F) \cup (E_2 - \pi_{\mathbf{B}}(E_2 \times F) = \emptyset \cup (E_2 - \emptyset) = E_2 .$$

Expressing "`if-then-else`" query in Relational Algebra (Alternative)

We begin with a special case of the "`if-then-else`" query, i.e.[2]

$$\texttt{if} \quad F \neq \emptyset \quad \texttt{then} \quad E_1$$
$$\texttt{else} \quad E_2$$

This query can also be expressed in RA with the expression

$$(E_1 \times \pi_{()}(F)) \cup (E_2 - (E_2 \times \pi_{()}(F))$$

[2] Recall that E_1 and E_2 both have schema **B**.

Expressing "`if-then-else`" query in Relational Algebra

$$(E_1 \times \pi_{()}(F)) \cup (E_2 - (E_2 \times \pi_{()}(F))$$

- If $F \neq \emptyset$, then $\pi_{()}(F) = \{()\}$.
  Thus,
  $$\begin{aligned} E_1 \times \pi()(F) &= E_1 \times \{()\} = E_1 \\ E_2 \times \pi_{()}(F) &= E_2 \times \{()\} = E_2. \end{aligned}$$
  Therefore,
  $$(E_1 \times \pi_{()}(F)) \cup (E_2 - (E_2 \times \pi_{()}(F)) = E_1 \cup (E_2 - E_2) = E_1 - \emptyset = E_1.$$
- If $F = \emptyset$, then $\pi_{()}(F) = \emptyset$. Therefore,
  $$(E_1 \times \pi_{()}(F)) \cup (E_2 - (E_2 \times \pi_{()}(F)) = \emptyset \cup (E_2 - \emptyset) = E_2.$$

Expressing "`if-then-else`" query in Relational Algebra

$$\texttt{if} \quad F \neq \emptyset \quad \texttt{then} \quad E_1$$
$$\texttt{else} \quad E_2$$

In RA,

$$\pi_{\mathbf{B}}(E_1 \times F) \cup (E_2 - \pi_{\mathbf{B}}(E_2 \times F)) \qquad (1)$$

or, alternatively,

$$(E_1 \times \pi_{()}(F)) \cup (E_2 - (E_2 \times \pi_{()}(F)) \qquad (2)$$

Expression (2) is better than expression (1):

- Complexity of expression (1) is $O((|E_1| + |E_2|) * |F|)$.
- Complexity of expression (2) is $O(|E_1| + |E_2| + |F|)$.

RA expression for "`if-then-else`" in SQL

$$\texttt{if} \quad F \neq \emptyset \quad \texttt{then} \quad E_1$$
$$\texttt{else} \quad E_2$$

$$\pi_{\mathbf{B}}(E_1 \times F) \cup (E_2 - \pi_{\mathbf{B}}(E_2 \times F)) \qquad (1)$$

```
SELECT    e1.*
FROM      E1 e1 CROSS JOIN F
UNION
(SELECT   e2.*
 FROM     E2 e2
 EXCEPT
 SELECT   e2.*
 FROM     E2 e2 CROSS JOIN F)
```

RA expression for "`if-then-else`" in SQL

$$\texttt{if} \quad F \neq \emptyset \quad \texttt{then} \quad E_1$$
$$\texttt{else} \quad E_2$$

$$(E_1 \times \pi_{()}(F)) \cup (E_2 - (E_2 \times \pi_{()}(F))) \qquad (2)$$

```
SELECT   e1.*
FROM     E1 e1 CROSS JOIN (SELECT DISTINCT row() FROM F) f
UNION
(SELECT  e2.*
 FROM    E2 e2
 EXCEPT
 SELECT  e2.*
 FROM    E2 e2 CROSS JOIN (SELECT DISTINCT row() FROM F) f )
```

Expressing "`if-then-else`" query in Relational Algebra

We next consider another special case of the "`if-then-else`" query, i.e.[3]

$$\texttt{if} \quad F = \emptyset \quad \texttt{then} \quad E_1$$
$$\texttt{else} \quad E_2$$

This query is equivalent with

$$\texttt{if} \quad F \neq \emptyset \quad \texttt{then} \quad E_2$$
$$\texttt{else} \quad E_1$$

This query can be expressed in RA with the expression

$$\pi_{\mathbf{B}}(E_2 \times F) \cup (E_1 - \pi_{\mathbf{B}}(E_1 \times F))$$

or, alternatively

$$(E_2 \times \pi_{()}(F)) \cup (E_1 - (E_1 \times \pi_{()}(F)).$$

[3] Recall that E_1 and E_2 both have schema **B**.

Expressing "`if-then-else`" query in Relational Algebra

We next consider another special case of the "`if-then-else`" query, i.e.

$$\texttt{if} \quad |F| \geq 2 \quad \texttt{then} \quad E_1 \\ \texttt{else} \quad E_2$$

This query is equivalent with[4]

$$\texttt{if} \quad (F_1 \bowtie_{F_1.\mathbf{A} \neq F_2.\mathbf{A}} F_2) \neq \emptyset \quad \texttt{then} \quad E_1 \\ \texttt{else} \quad E_2$$

This query can be expressed in RA with the expression

$$\pi_{\mathbf{B}}(E_2 \times (F_1 \bowtie_{F_1.\mathbf{A} \neq F_2.\mathbf{A}} F_2)) \cup (E_1 - \pi_{\mathbf{A}}(E_1 \times (F_1 \bowtie_{F_1.\mathbf{A} \neq F_2.\mathbf{A}} F_2)))$$

or, alternatively

$$E_2 \times \pi_{()}(F_1 \bowtie_{F_1.\mathbf{A} \neq F_2.\mathbf{A}} F_2) \cup (E_1 - (E_1 \times \pi_{()}(F_1 \bowtie_{F_1.\mathbf{A} \neq F_2.\mathbf{A}} F_2)).$$

---

[4] Recall that F has schema $\mathbf{A}$.

Boolean queries

A boolean query is a special case of the "`if-then-else`" query. I.e., it is the following query:

```
if  C(F)  then  true
          else  false
```

F	RA expression with schema **A**
$\mathcal{C}(F)$	a boolean set/relation condition $\mathcal{C}$ on F

Typical cases for $\mathcal{C}(F)$:

$$F \neq \emptyset$$
$$F = \emptyset$$
$$|F| \;\theta\; \mathrm{k} \quad \text{with } \theta \text{ one of } =, \neq, <, \leq, >, \geq$$

Semantics of boolean queries

The semantics of the boolean query

$$\texttt{if} \quad \mathcal{C}(F) \quad \begin{array}{ll} \texttt{then} & \texttt{true} \\ \texttt{else} & \texttt{false} \end{array}$$

If $\mathcal{C}(F)$ is true then the boolean query returns the value "`true`".

If $\mathcal{C}(F)$ is false then the boolean query returns the value "`false`".

Expressing boolean queries in Relational Algebra

We begin with a special case of a boolean query, i.e:

$$\texttt{if} \quad F \neq \emptyset \quad \texttt{then} \quad \texttt{true}$$
$$\texttt{else} \quad \texttt{false}$$

This query can be expressed in RA with the expression

$$\pi_B((B:\texttt{true}) \times F) \cup ((B:\texttt{false}) - \pi_B((B:\texttt{false}) \times F))$$

or, alternatively

$$(B:\texttt{true}) \times \pi_{()}(F) \cup ((B:\texttt{false}) - ((B:\texttt{false}) \times \pi_{()}(F))).$$

Recall that $(B:\texttt{true})$ and $(B:\texttt{false})$ are RA expressions representing the constants "true" and "false", respectively.

Expressing boolean queries in Relational Algebra (complexity)

$$\texttt{if} \quad F \neq \emptyset \quad \texttt{then} \quad \texttt{true}$$
$$\texttt{else} \quad \texttt{false}$$

In RA,

$$\pi_B((B : \texttt{true}) \times F) \cup ((B : \texttt{false}) - \pi_B((B : \texttt{false}) \times F)) \quad (1)$$

or, alternatively,

$$(B : \texttt{true}) \times \pi_{()}(F) \cup ((B : \texttt{false}) - ((B : \texttt{false}) \times \pi_{()}(F))) \quad (2)$$

Expressions (1) and (2) have the same complexity, namely $O(|F|)$.

RA expression for boolean RA queries in SQL

$$\texttt{if} \quad F \neq \emptyset \quad \texttt{then} \quad \texttt{true}$$
$$\texttt{else} \quad \texttt{false}$$

$$\pi_B((B : \texttt{true}) \times F) \cup ((B : \texttt{false}) - \pi_B((B : \texttt{false}) \times F))$$

```
SELECT t.B
FROM   (SELECT true AS B)  t CROSS JOIN F
UNION
(SELECT false AS B
 EXCEPT
 SELECT f.B
 FROM   (SELECT false AS B)  f CROSS JOIN F)
```

Application of Boolean RA query

Let R be a binary relation over schema (A, B).
We say that R is a *function* from A to B if

$$\forall t_1 \forall t_2((R(t_1) \wedge R(t_2) \wedge t_1.A = t_2.B) \rightarrow t_1.B = t_2.B)$$

Or, equivalently

$$\neg \exists t_1 \exists t_2 (R(t_1) \wedge R(t_2) \wedge t_1.A = t_2.B \wedge t_1.B \neq t_2.B).$$

Consider the subformula

$$(R(t_1) \wedge R(t_2) \wedge t_1.A = t_2.B \wedge t_1.B \neq t_2.B).$$

The RA expression for this subformula is

$$R_1 \bowtie_{R_1.A=R_2.A \wedge R_1.B \neq R_2.B} R_2.$$

Application of Boolean RA query

Let R be a binary relation over schema (A, B).

R is a function if

$$\neg\exists t_1 \exists t_2 (R(t_1) \wedge R(t_2) \wedge t_1.A = t_2.B \wedge t_1.B \neq t_2.B).$$

We can express this as the RA boolean query:

$$\texttt{if} \quad (R_1 \bowtie_{R_1.A = R_2.A \wedge R_1.B \neq R_2.B} R_2) = \emptyset \quad \begin{array}{ll} \texttt{then} & \texttt{true} \\ \texttt{else} & \texttt{false} \end{array}$$

This boolean RA query can then be expressed in RA and SQL with RA operations as shown above.

Application of Boolean RA query

Let R be a binary relation over schema (A, B).

The boolean query "R is a function" can be expressed in RA as

$$\pi_B((B : \texttt{true}) \times F) \cup ((B : \texttt{false}) - \pi_B((B : \texttt{false}) \times F))$$

where F is the RA expression

$$R_1 \bowtie_{R_1.A = R_2.A \wedge R_1.B \neq R_2.B} R_2.$$

Application of Boolean RA query

Let *R* be a binary relation over schema (*A*, *B*).
The boolean query "*R* is a function" can be expressed in SQL with RA operations as

```
WITH F AS
    (SELECT t1.*, t2.*
     FROM   R t1 JOIN R t2 ON (t1.A=t2.A AND t1.B <> t2.B))
SELECT t.B
FROM   (SELECT false AS B) t CROSS JOIN F
UNION
(SELECT true AS B
 EXCEPT
 SELECT f.B
 FROM   (SELECT true AS B) f CROSS JOIN F)
```

Translating Boolean SQL queries to Boolean RA queries

Consider the boolean SQL query[5]

SELECT EXISTS (Q)

Let E_Q denote the translation of “Q" in the Relational Algebra. Then the boolean SQL query can be expressed as the RA boolean query:

$$\texttt{if} \quad E_Q \neq \emptyset \quad \texttt{then} \quad \texttt{true}$$
$$\texttt{else} \quad \texttt{false}$$

I.e., as the RA expression

$$(B:\texttt{true})\times\pi_{()}(E_Q)\cup((B:\texttt{false})-(B:\texttt{false})\times\pi_{()}(E_Q))$$

[5] “Q" denotes a SQL query.

Translating Boolean SQL queries to Boolean RA queries

Consider the boolean SQL query[6]

SELECT NOT EXISTS (Q)

Let E_Q denote the translation of “Q" in the Relational Algebra. Then the boolean SQL query can be expressed as the RA boolean query:

$$\begin{array}{llll} \texttt{if} & E_Q = \emptyset & \texttt{then} & \texttt{true} \\ & & \texttt{else} & \texttt{false} \end{array}$$

I.e., as the RA expression

$$(B : \texttt{false}) \times \pi_{()}(E_Q) \cup ((B : \texttt{true}) - (B : \texttt{true}) \times \pi_{()}(E_Q))$$

[6] “Q" denotes a SQL query.