On the Correspondence Between the Tuple Relational Calculus (TRC) and SQL (Draft)

Dirk Van Gucht[1]

[1] Indiana University

Learning outcomes

- Pure SQL is a formal logic:
  - It can express statements (constraints–boolean queries)
  - It can express set (relation) abstractions (queries)
- Pure SQL corresponds in a one-to-one manner with Tuple Relation Calculus (a variant of Predicate Logic):[1]
  - The semantics of TRC provides the semantics for Pure SQL: this is at the basis of correct query evaluation and constraint verification
  - The proof theory of TRC provides techniques to rewrite SQL queries into equivalent queries: this is at the basis of query optimization which is essential for efficient query evaluation and constaint verification

[1] More accurately, Pure SQL is equivalent with *SafeTuple Relation Calculus*, a logic that places some restrictions on the free tuples in TRC formulas.

Overview

- Tuple Relational Calculus (TRC) syntax
- Tuple Relation Calculus (TRC) constraints
- Tuple Relational Calculus (TRC) queries
- Expressing TRC constraints as Boolean SQL queries
- Expressing TRC queries as SQL queries

Database Model for TRC

The database model for TRC is the *Relational Database Model*. Formally,

- a set of *attributes* $\mathcal{A}$;
- each attribute A has as an associated *domain* $\mathbf{dom}(A)$.

- a set of *relation schemas* $\mathcal{R}$;
- each relation schema R has an associated list of attributes $\mathbf{att}(R)$.

- a *relational database schema* $\mathbf{R}$ is a sequence of relation schemas $R_1, R_2, \ldots, R_n$:
  - $\mathbf{R} = (R_1, \ldots, R_n)$.

Example: database schema

$\texttt{Student}(\underline{\textit{sid}} : \texttt{integer}, \textit{sname} : \texttt{text})$

$\texttt{Book}(\underline{\textit{bookno}} : \texttt{integer}, \textit{title} : \texttt{text}, \textit{price} : \texttt{integer})$

$\texttt{Buys}(\underline{\textit{sid} : \texttt{integer}, \textit{bookno} : \texttt{integer}})$

TRC terms

Terms serve to represent objects in TRC formulas.

- A *tuple variable* t has an associated sequence of attributes $\mathbf{att}(t)$ and will represent a tuple object with components across this sequence of attributes;
- If a tuple variable t has $\mathbf{att}(t) = (A_1, \ldots, A_k)$, then t has a sequence of associated *tuple variable components* denoted by $t.A_1, \ldots, t.A_k$, and these represent objects in $\mathbf{dom}(A_1)$ through $\mathbf{dom}(A_k)$ respectively; and
- for each domain value of an attribute there is a *constant* that names this domain value.

TRC Formulas $\mathcal{F}(\mathbf{R})$ (Syntax)

$R(t)$	$\mathbf{att}(R) = \mathbf{att}(t)$	
$t.A\,\theta\,s.B$	$\mathbf{dom}(A) = \mathbf{dom}(B)$	$\theta := =, \neq, <, \leq, >,$ or $\geq$
$t.A\,\theta\,\mathbf{a}$	$\mathbf{a} \in \mathbf{dom}(A)$	$\theta := =, \neq, <, \leq, >,$ or $\geq$
$F_1 \wedge F_2$	conjunction (and)	
$F_1 \vee F_2$	disjunction (or)	
$F_1 \rightarrow F_2$	implication (if then)	
$\neg F$	negation (not)	
$\exists\, t\, F$	existential quantification	
$\forall\, t\, F$	universal quantification	
(F)		

Precedence order ($\succ$): $\neg \succ \wedge \succ \vee \succ \rightarrow$

TRC Formulas $\mathcal{F}(\mathbf{R})$ in abbreviated form (Syntax)

$R(t)$	$\mathbf{att}(R) = \mathbf{att}(t)$	
$t.A\,\theta\, s.B$	$\mathbf{dom}(A) = \mathbf{dom}(B)$	$\theta :=, \neq, <, \leq, >,$ or $\geq$
$t.A\,\theta\,\mathbf{a}$	$\mathbf{a} \in \mathbf{dom}(A)$	$\theta :=, \neq, <, \leq, >,$ or $\geq$
$F_1 \wedge F_2$	conjunction (and)	
$F_1 \vee F_2$	disjunction (or)	
$F_1 \rightarrow F_2$	implication (if then)	
$\neg F$	negation (not)	
$\exists\, t\, F$	existential quantification	
$\forall\, t\, F$	universal quantification	
$\exists\, t_1 \in R_1, \ldots, t_k \in R_k(F)$	series of existential quantifiers	
$\forall\, t_1 \in R_1, \ldots, t_k \in R_k(F \rightarrow G)$	series of universal quantifiers	
(F)		

TRC Formulas $\mathcal{F}(\mathbf{R})$ in abbreviated form (Syntax)

$$\exists\, t_1 \in R_1, \ldots, t_k \in R_k (F)$$

is an abbreviated form for the formula

$$\exists t_1 \ldots \exists t_k (R_1(t_1) \wedge \cdots \wedge R_k(t_k) \wedge F)$$

$$\forall\, t_1 \in R_1, \ldots, t_k \in R_k (F \rightarrow G)$$

is an abbreviated form for the formula

$$\forall t_1 \ldots \forall t_k ((R_1(t_1) \wedge \cdots \wedge R_k(t_k) \wedge F) \rightarrow G)$$

Formulas (Examples)

$\texttt{Student}(s)$

$\texttt{Student}(s) \wedge s.sname = \textbf{Eric}$

$\exists s(\texttt{Student}(s) \wedge s.sname = \textbf{Eric})$

$\texttt{Student}(s) \wedge \exists t(\texttt{Buys}(t) \wedge s.sname = \textbf{Eric} \wedge t.sid = s.sid)$

$\exists s \exists t(\texttt{Student}(s) \wedge \texttt{Buys}(t) \wedge s.sname = \textbf{Eric} \wedge t.sid = s.sid)$

$\exists s(\texttt{Student}(s) \wedge \exists t(\texttt{Buys}(t) \wedge s.sname = \textbf{Eric} \wedge t.sid = s.sid))$

$\exists s(\texttt{Student}(s) \wedge \forall t((\texttt{Buys}(t) \wedge t.sid = s.sid) \rightarrow t.bookno \neq 1000)$

Formulas in abbreviated form

$\texttt{Student}(s)$

$\texttt{Student}(s) \wedge s.sname = \textbf{Eric}$

$\exists s \in \texttt{Student}(s.sname = \textbf{Eric})$

$\texttt{Student}(s) \wedge \exists t \in \texttt{Buys}(s.sname = \textbf{Eric} \wedge t.sid = s.sid)$

$\exists s \in \texttt{Student}, t \in \texttt{Buys}(s.sname = \textbf{Eric} \wedge t.sid = s.sid)$

$\exists s \in \texttt{Student}(\exists t \in \texttt{Buys}(s.sname = \textbf{Eric} \wedge t.sid = s.sid))$

$\exists s \in \texttt{Student}(\forall t \in \texttt{Buys}(t.sid = s.sid \rightarrow t.bookno \neq 1000))$

Free and bound variables of TRC formulas

A tuple variable is *bound* if gets bound by an associated existential or universal *quantifier*. It is *free* otherwise.[2]

TRC Formula H	**free**(H)	**bound**(H)
$R(t)$	$\{t\}$	$\{\}$
$t.A\,\theta\,s.B$	$\{t, s\}$	$\{\}$
$t.A\,\theta\,\mathbf{a}$	$\{t\}$	$\{\}$
$F_1 \wedge F_2$	$\mathbf{free}(F_1) \cup \mathbf{free}(F_2)$	$\mathbf{bound}(F_1) \cup \mathbf{bound}(F_2)$
$F_1 \vee F_2$	$\mathbf{free}(F_1) \cup \mathbf{free}(F_2)$	$\mathbf{bound}(F_1) \cup \mathbf{bound}(F_2)$
$F_1 \rightarrow F_2$	$\mathbf{free}(F_1) \cup \mathbf{free}(F_2)$	$\mathbf{bound}(F_1) \cup \mathbf{bound}(F_2)$
$\neg F$	$\mathbf{free}(F)$	$\mathbf{bound}(F)$
$\exists\, t\, F$	$\mathbf{free}(F) - \{t\}$	$\mathbf{bound}(F) \cup \{t\}$
$\forall\, t\, F$	$\mathbf{free}(F) - \{t\}$	$\mathbf{bound}(F) \cup \{t\}$
$\exists\, t_1 \in R_1, \ldots, t_k \in R_k(F)$	$\mathbf{free}(F) - \{t_1, \ldots, t_k\}$	$\mathbf{bound}(F) \cup \{t_1, \ldots, t_k\}$
$\forall\, t_1 \in R_1, \ldots, t_k \in R_k(F \rightarrow G)$	$\mathbf{free}(F \rightarrow G) - \{t_1, \ldots, t_k\}$	$\mathbf{bound}(F \rightarrow G) \cup \{t_1, \ldots, t_k\}$
(F)	$\mathbf{free}(F)$	$\mathbf{bound}(F)$

$F(R_1, \ldots, R_n; t_1, \ldots, t_k)$:

- R_1 through R_n are relation schemas appearing in formula F; and
- t_1 through t_k are the free variables of formula F.

[2] For an SQL query, free variables are exclusively those that are introduced in the outermost `FROM` clause of the query.

Formula and its free and bound variables (Examples)

Formula	**free**	**bound**
$\texttt{Student}(s)$	$\{s\}$	$\{\}$
$\texttt{Student}(s) \wedge s.sname = \textbf{Eric}$	$\{s\}$	$\{\}$
$\exists\, s(\texttt{Student}(s) \wedge s.sname = \textbf{Eric})$	$\{\}$	$\{s\}$
$\texttt{Student}(s) \wedge \exists\, t \in \texttt{Buys}(\textbf{Eric} \wedge t.sid = s.sid)$	$\{s\}$	$\{t\}$
$\exists\, s\, \exists\, t(\texttt{Student}(s) \wedge \texttt{Buys}(t) \wedge s.sname = \textbf{Eric} \wedge t.sid = s.sid)$	$\{\}$	$\{s, t\}$
$\exists\, s(\texttt{Student}(s) \wedge \exists\, t(\texttt{Buys}(t) \wedge s.sname = \textbf{Eric} \wedge t.sid = s.sid))$	$\{\}$	$\{s, t\}$
$\exists\, s(\texttt{Student}(s) \wedge \forall\, t(\texttt{Buys}(t) \wedge t.sid = s.sid \rightarrow t.bookno \neq 1000))$	$\{\}$	$\{s, t\}$

Fundamental equivalence involving implication and universal quantifcation

Truth table of $F \rightarrow G$:

F	G	$F \rightarrow G$	$\neg F \vee G$	$\neg(F \wedge \neg G)$
t	t	t	t	t
t	f	f	f	f
f	t	t	t	t
f	f	t	t	t

$$\begin{aligned}
F \rightarrow G \quad & \Leftrightarrow \quad \neg F \vee G \\
& \Leftrightarrow \quad \neg(F \wedge \neg G)
\end{aligned}$$

$$\begin{aligned}
\neg(F \rightarrow G) \quad & \Leftrightarrow \quad \neg(\neg(F \wedge \neg G)) \\
& \Leftrightarrow \quad F \wedge \neg G
\end{aligned}$$

$$\begin{aligned}
\forall t_1 \in R_1, \ldots t_k \in R_k (F \rightarrow G) \quad & \Leftrightarrow \quad \neg \exists t_1 \in R_1, \ldots t_k \in R_k \neg(F \rightarrow G) \\
& \Leftrightarrow \quad \neg \exists t_1 \in R_1, \ldots t_k \in R_k (F \wedge \neg G)
\end{aligned}$$

TRC constraints

A formula F is a *constraint* if **free**$(F) = \{\}$.

A constraint is a *statement* that is either true or false.

Primary keys and foreign keys are examples of constraints.

TRC constraints (Examples)

There is a student whose name is Eric
$\exists\, s(\texttt{Student}(s) \wedge s.sname = \textbf{Eric})$

There is a student whose name is Eric and who bought a book
$\exists\, s(\texttt{Student}(s) \wedge s.sname = \textbf{Eric} \wedge \exists\, t(\texttt{Buys}(t) \wedge t.sid = s.sid))$

There is a student who only bought books with bookno $\neq$ 1000
$\exists\, s(\texttt{Student}(s) \wedge \forall\, t((\texttt{Buys}(t) \wedge t.sid = s.sid) \rightarrow t.bookno \neq 1000))$

sid is a primary key for the Student relation
$$\forall s_1 \forall s_2((\texttt{Student}(s_1) \wedge \texttt{Student}(s_2) \wedge s_1.sid = s_2.sid) \rightarrow s_1.sname = s_2.sname)$$

TRC constraints in abbreviated form (Examples)

There is a student whose name is Eric

$\exists\, s \in \texttt{Student}\,(s.sname = \textbf{Eric})$

There is a student whose name is Eric and who bought a book

$\exists\, s \in \texttt{Student}(s.sname = \textbf{Eric} \land \exists\, t \in \texttt{Buys}(t.sid = s.sid))$

There is a student who only bought books with bookno $\neq$ 1000

$\exists\, s \in \texttt{Student}(\forall\, t \in \texttt{Buys}(t.sid = s.sid \rightarrow t.bookno \neq 1000))$

sid is a primary key for the Student relation

$\forall s_1 \in \texttt{Student}, s_2 \in \texttt{Student}(s_1.sid = s_2.sid \rightarrow s_1.sname = s_2.sname)$

TRC constraints expressed as boolean SQL queries

TRC constraint:

There is a student whose name is Eric

$$\exists\, s \in \texttt{Student}\,(s.sname = \textbf{Eric})$$

TRC constraint using $\neq \emptyset$ set predicate:

$$\{s.sid, s.sname \mid \texttt{Student}(s) \land s.sname = \textbf{Eric})\} \neq \emptyset$$

Boolean SQL query:

```
select exists(select s.sid, s.sname
              from   Student s
              where  s.sname = 'Eric');
```

TRC constraints expressed as boolean SQL queries

TRC constraint:

There is a student whose name is Eric

$$\exists\, s \in \texttt{Student}\,(s.sname = \textbf{Eric})$$

TRC constraint using $\neq \emptyset$ set predicate:

$$\{1 \mid \texttt{Student}(s) \wedge s.sname = \textbf{Eric}\} \neq \emptyset$$

Boolean SQL query:

```
select exists(select 1
              from   Student s
              where  s.sname = 'Eric');
```

TRC constraints expressed as boolean SQL queries

TRC constraint:

There is a student whose name is Eric and who bought a book.

$$\exists\, s \in \texttt{Student}(s.sname = \textbf{Eric} \land \exists\, t \in \texttt{Buys}(t.sid = s.sid))$$

Boolean SQL query:

```
select exists(select 1
              from   Student s
              where  s.sname = 'Eric' and
                     exists(select 1
                            from   Buys t
                            where  t.sid = s.sid));
```

TRC constraints expressed as boolean SQL queries

TRC constraint:

There is a student whose name is Eric and who bought a book.

$$\exists\, s \in \texttt{Student}, t \in \texttt{Buys}(s.sname = \textbf{Eric} \wedge t.sid = s.sid)$$

Boolean SQL query:

```
select exists(select 1
              from   Student s, Buys t
              where  s.sname = 'Eric' and t.sid = s.sid)
```

TRC constraints expressed as boolean SQL queries

There is a student who only bought books with bookno $\neq 1000$
$\exists s \in \texttt{Student}(\forall t \in \texttt{Buys}(t.sid = s.sid \rightarrow t.bookno \neq 1000))$

Equivalently,
$\exists s \in \texttt{Student}(\neg\exists t \in \texttt{Buys}\,\neg(t.sid = s.sid \rightarrow t.bookno \neq 1000))$

Equivalently,
$\exists s \in \texttt{Student}(\neg\exists t \in \texttt{Buys}\,(t.sid = s.sid \wedge \neg(t.bookno \neq 1000)))$

Equivalently,
$\exists s \in \texttt{Student}(\neg\exists t \in \texttt{Buys}\,(t.sid = s.sid \wedge t.bookno = 1000))$

$$\{1 \mid \texttt{Student}(s) \wedge \\ \neg\{1 \mid \texttt{Buys}(t) \wedge t.sid = s.sid \wedge t.bookno = 1000\} \neq \emptyset\} \neq \emptyset$$

TRC constraints expressed as boolean SQL queries

There is a student who only bought books with bookno $\neq$ 1000

$$\exists\, s \in \texttt{Student}(\forall\, t \in \texttt{Buys}(t.sid = s.sid \rightarrow t.bookno \neq 1000))$$

$$\{1 \mid \texttt{Student}(s) \wedge \\ \neg\{1 \mid t \in \texttt{Buys}(t.sid = s.sid \wedge t.bookno = 1000)\} \neq \emptyset\} \neq \emptyset$$

Boolean SQL query:

```
select exists(select 1
              from   Student s
              where  not exists(select 1
                                from   Buys t
                                where  t.sid = s.sid and
                                       t.bookno = 1000));
```

TRC constraints expressed as boolean SQL queries

sid is a primary key for the Student relation

$$\forall s_1 \in \texttt{Student}, s_2 \in \texttt{Student}(s_1.sid = s_2.sid \rightarrow s_1.sname = s_2.sname)$$

Equivalently,

$$\neg\exists s_1 \in \texttt{Student}, s_2 \in \texttt{Student}(s_1.sid = s_2.sid \wedge \neg(s_1.sname = s_2.sname))$$

Equivalently,

$$\neg\exists s_1 \in \texttt{Student}, s_2 \in \texttt{Student}(s_1.sid = s_2.sid \wedge s_1.sname \neq s_2.sname)$$

Boolean SQL query:

```
select not exists(select 1
                  from   Student s1, Student s2
                  where  s1.sid = s2.sid and
                         s1.sname <> s2.sname);
```

TRC constraints expressed as boolean SQL queries

Each student bought at least two books.

$$\forall s \in \texttt{Student}(\exists t_1 \in \texttt{Buys}, t_2 \in \texttt{Buys}(t_1.sid = s.sid \wedge t_2.sid = s.sid \wedge t_1.bookno \neq t_2.bookno))$$

Equivalently,

$$\neg\exists s \in \texttt{Student}\neg(\exists t_1 \in \texttt{Buys}, t_2 \in \texttt{Buys}(t_1.sid = s.sid \wedge t_2.sid = s.sid \wedge t_1.bookno \neq t_2.bookno))$$

Boolean SQL query:

```
select not exists(select 1
                  from   Student s
                  where  not exists(select 1
                                    from   Buys  t1, Buys t2
                                    where  t1.sid = s.sid and
                                           t2.sid = s.sid and
                                           t1.bookno <> t2.bookno));
```

TRC queries

A *TRC query* over a database schema **R** is a set-abstraction of the form

$$\{(t_{j_1}.A_1, \ldots, t_{j_m}.A_m) | F(R_1, \ldots, R_n; t_1, \ldots, t_k)\}$$

where

1. F is a TRC formula in $\mathcal{F}(\mathbf{R})$,
2. $\{t_{j_1}, \ldots, t_{j_m}\} \subseteq \mathbf{free}(F)$, and
3. for $i \in [1, m]$, $A_i \in \mathbf{att}(t_{j_i})$.

We will call this the query F.

The schema associated with F is $(A_1, \ldots, A_m)$, and denote it by $\mathbf{att}(F)$.

TRC queries (Examples)

$$\{(s.id, s.name) \mid \texttt{Student}(s)\}$$

$$\{(s.sid) \mid \texttt{Student}(s) \wedge s.sname = \textbf{Eric}\}$$

$$\{() \mid \exists\, s \in \texttt{Student}(s.sname = \textbf{Eric})\}^{3}$$

$$\{(s.sid, t.bookno) \mid \texttt{Student}(s) \wedge s.sname = \textbf{Eric} \wedge \texttt{Buys}(t) \wedge t.sid = s.sid)\}$$

$$\{(s.sid) \mid \texttt{Student}(s) \wedge \forall\, t \in \texttt{Buys}(t.sid = s.sid \rightarrow t.bookno \neq 1000))\}$$

[3] () denotes the empty tuple, i.e., the tuple without components

TRC queries expressed as SQL queries

Find the sid and name of each student.

$\{(s.sid, s.sname) \mid \texttt{Student}(s)\}$

SQL query:

```
select s.sid, s.sname
from   Student s;
```

TRC queries expressed as SQL queries

Find the sid of each student whose name is Eric.

$$\{(s.sid) \mid \texttt{Student}(s) \wedge s.sname = \textbf{Eric}\}$$

SQL query:

```
select s.sid
from   Student s
where  s.sname = 'Eric';
```

TRC queries expressed as SQL queries

Witness if there is a student whose name is Eric.

$$\{() \mid \exists \boldsymbol{s} \in \texttt{Student}(s.sname = \textbf{Eric})\}$$

SQL query:

```
select  row()
from    Student s
where   s.sname = 'Eric';
```

TRC queries expressed as SQL queries

Find the (s,b) pairs of each student with sid s and name Eric who bought a book with bookno *b*.

$$\{(s.sid, t.bookno) \mid \texttt{Student}(s) \wedge s.sname = \textbf{Eric} \wedge \texttt{Buys}(t) \wedge t.sid = s.sid\}$$

SQL query:

```
select s.sid, t.bookno
from   Student s, Buys t
where  s.sname = 'Eric' and
       t.sid = s.sid;
```

TRC queries expressed as SQL queries using IN predicate

Find the sid of each student whose name is Eric and who bought a book.

$$\{(s.sid) \mid \texttt{Student}(s) \wedge s.sname = \textbf{Eric} \wedge \exists t \in \texttt{Buys}(t.sid = s.sid)\}$$

Equivalently,

$$\{(s.sid) \mid \texttt{Student}(s) \wedge s.sname = \textbf{Eric} \wedge s.sid \in \{t.sid \mid \texttt{Buys}(t)\}\}$$

SQL query:

```
select s.sid
from   Student s
where  s.sname = 'Eric' and
       s.sid in (select t.sid
                 from   Buys t)
```

TRC queries expressed as SQL queries

Find the sid of each student who only bought books except that with bookno 1000.

$$\{(s.sid) \mid \texttt{Student}(s) \land \forall t \in \texttt{Buys}(t.sid = s.sid \rightarrow t.bookno \neq 1000))\}$$

Or, equivalently

$$\{(s.sid) \mid \texttt{Student}(s) \land \neg\exists t \in \texttt{Buys}(t.sid = s.sid \land t.bookno = 1000)\}$$

SQL query:

```
select s.sid
from   Student s
where  not exists(select 1
                  from   Buys t
                  where  t.sid = s.sid and
                         t.bookno = 1000);
```

TRC queries expressed as SQL queries using NOT IN predicate

Find the sid of each student who only bought books except that with bookno 1000.

$$\{(s.sid) \mid \texttt{Student}(s) \land \forall t \in \texttt{Buys}(t.sid = s.sid \rightarrow t.bookno \neq 1000))\}$$

SQL query:

```
select s.sid
from   Student s
where  not exists(select 1
                  from   Buys t
                  where  t.sid = s.sid and
                         t.bookno = 1000);
```

Or, using `NOT IN` set membership predicate

```
select s.sid
from   Student s
where  s.sid not in (select t.sid
                     from   Buys t
                     where  t.bookno = 1000);
```

Logical Equivalences involving formulas F, G, and H

$\neg\neg F$	$\equiv$	F	Double Negation
$\neg(F \wedge G)$	$\equiv$	$\neg F \vee \neg G$	De Morgan for $\wedge$
$\neg(F \vee G)$	$\equiv$	$\neg F \wedge \neg G$	De Morgan for $\vee$
$F \wedge (G \vee H)$	$\equiv$	$(F \wedge G) \vee (F \wedge H)$	Distribution of $\wedge$ over $\vee$
$F \vee (G \wedge H)$	$\equiv$	$(F \vee G) \wedge (F \vee H)$	Distribution of $\vee$ over $\wedge$
$F \wedge F$	$\equiv$	F	Idempotence of $\wedge$
$F \vee F$	$\equiv$	F	Idempotence of $\vee$
$F \wedge (F \vee G)$	$\equiv$	F	Absorption Law for $\wedge$
$F \vee (F \wedge G)$	$\equiv$	F	Absorption Law for $\vee$
$F \wedge G$	$\equiv$	$G \wedge F$	Commutativity of $\wedge$
$F \vee G$	$\equiv$	$G \vee F$	Commutativity of $\vee$
$F \wedge (G \wedge H)$	$\equiv$	$(F \wedge G) \wedge H$	Associativity of $\wedge$
$F \vee (G \vee H)$	$\equiv$	$(F \vee G) \vee H$	Associativity of $\vee$
$F \rightarrow G$	$\equiv$	$\neg F \vee G$	Conditional Law
$F \rightarrow G$	$\equiv$	$\neg G \rightarrow \neg F$	Contrapositive Law
$\neg(F \rightarrow G)$	$\equiv$	$F \wedge \neg G$	Abjunction Law
$F \rightarrow (F \wedge G)$	$\equiv$	$F \rightarrow G$	Absorption Law of Conditional
$(F \wedge G) \rightarrow H$	$\equiv$	$F \rightarrow (G \rightarrow H)$	Exportation Law of Conditional
$(F \wedge G) \rightarrow H$	$\equiv$	$(F \rightarrow H) \vee (G \rightarrow H)$	Left Distribution of $\wedge$ over $\rightarrow$
$(F \vee G) \rightarrow H$	$\equiv$	$(F \rightarrow H) \wedge (G \rightarrow H)$	Left Distribution of $\vee$ over $\rightarrow$
$F \rightarrow (G \wedge H)$	$\equiv$	$(F \rightarrow G) \wedge (F \rightarrow H)$	Right Distribution of $\wedge$ over $\rightarrow$
$F \rightarrow (G \vee H)$	$\equiv$	$(F \rightarrow G) \vee (F \rightarrow H)$	Right Distribution of $\vee$ over $\rightarrow$

Logical Equivalences involving formulas F and G and truth formulas true and false

$\neg$ **true**	$\equiv$	**false**	
$\neg$ **false**	$\equiv$	**true**	
$F \wedge \neg F$	$\equiv$	**false**	Contradiction Law for $\wedge$
$F \vee \neg F$	$\equiv$	**true**	Tautology Law for $\vee$
$F \wedge$ **true**	$\equiv$	F	Identity Law for $\wedge$
$F \vee$ **false**	$\equiv$	F	Identity Law for $\vee$
$F \wedge$ **false**	$\equiv$	**false**	Domination Law for $\wedge$
$F \vee$ **true**	$\equiv$	**true**	Domination Law for $\vee$
false $\rightarrow F$	$\equiv$	**true**	Left Domination Law for $\rightarrow$
true $\rightarrow F$	$\equiv$	F	Right Identity Law for $\rightarrow$
$F \rightarrow$ **true**	$\equiv$	**true**	Right Domination Law for $\rightarrow$
$F \rightarrow$ **false**	$\equiv$	$\neg F$	Left Identity Law for $\rightarrow$
$F \rightarrow F$	$\equiv$	**true**	Domination Law for $\rightarrow$
$((F \rightarrow G) \rightarrow F) \rightarrow F$	$\equiv$	**true**	Pierce's Law

Logical equivalences for formulas with quantifiers

$\neg \exists t\, F(t)$	$\equiv$	$\forall t\ \neg F(t)$	De Morgan for $\exists$
$\neg \forall t\, F(t)$	$\equiv$	$\exists t\ \neg F(t)$	De Morgan for $\forall$
$\exists t\, F(t)$	$\equiv$	$\neg \forall t\ \neg F(t)$	
$\forall t\, F(t)$	$\equiv$	$\neg \exists t\ \neg F(t)$	
$\exists t\, (F(t) \vee G(t))$	$\equiv$	$(\exists t\, F(t)) \vee (\exists t\, G(t))$	Distribution of $\exists$ over $\vee$
$\forall t\, (F(t) \wedge G(t))$	$\equiv$	$(\forall t\, F(t)) \wedge (\forall t\, G(t))$	Distribution of $\forall$ over $\wedge$
$\exists t\, (F \wedge G(t))$	$\equiv$	$F \wedge (\exists t\, G(t))$	if t is not free in F
$\forall t\, (F \vee G(t))$	$\equiv$	$F \vee (\forall t\, G(t))$	if t is not free in F
$\exists t\, F$	$\equiv$	F	if t is not free in F
$\forall t\, F$	$\equiv$	F	if t is not free in F
$\exists t\, (F(t) \wedge G(t))$	$\equiv$	$\neg \forall t\, (F(t) \rightarrow \neg G(t))$	
$\forall t\, (F(t) \rightarrow G(t))$	$\equiv$	$\neg \exists t\, (F(t) \wedge \neg G(t))$	
$\exists t_1\, \exists t_2\, F(t_1, t_2)$	$\equiv$	$\exists t_2\, \exists t_1\, F(t_1, t_2)$	
$\forall t_1\, \forall t_2\, F(t_1, t_2)$	$\equiv$	$\forall t_2\, \forall t_1\, F(t_1, t_2)$	