Abstract. In this paper, we present some results on the equivalence between SQL queries and relation calculus expressions. These results are used to translate SQL queries into equivalent expressions of relational algebra.

I. INTRODUCTION

Stefano Ceri and Georg Gottlob, in [1] presented a translator from a subset of SQL queries into relational algebra. We have extended the results in [1] for subqueries with GROUP BY clause which can be nested at any level of a SQL query, and some extensions to condition of HAVING clause. This work is a basis step for this translation. In this paper, we have proved some results on the equivalence between SQL query and relational calculus expression.

II. BASIS KNOWLEDGE

1. Relational calculus

A relational calculus expression is of the form

\[ \{t((\text{components}))|\psi(t)\}, \]

where

- \( t \) is a tuple variable;
- \( \text{components} \) is a list of components of the form:
  - \( A_i \) - is an attribute,
  - \( R.A_i \) - \( R \) is a relation name, \( A_i \) is an attribute of \( R \);
- \( \psi(t) \) is a formula building from the atoms and collection of logical operators.

In order to use aggregate function in relational calculus, we extended the \( \text{components} \) to accept the form \( F_i[A_j] \), where \( F_i \) is a function and \( A_j \) is an attribute, and other extensions to the atoms of formula \( \psi \) of some types such that: \( F_i[A_j](s) \Theta u[A_i] \), \( F_i[A_j](s) \Theta a \), where \( F_i[A_j](s) \) is value of function \( F_i \) computed on attribute \( A_j \) for the tuple \( s \), \( a \) is a constant, and \( \Theta \in \{=, \neq, >, \geq, <, \leq\} \).

2. Structured Query Language (SQL)

a. Syntax of SQL query

\[
\begin{align*}
\text{SELECT} & \ (\text{selector})^* \ \text{FROM} \ (\text{relation.list}) \ \text{WHERE} \ (\text{predicate}) \\
& \ [\text{GROUP BY} \ (\text{gb.attr}) \ \text{HAVING} \ (\text{hav.condition})] \\
\end{align*}
\]

b. The meaning of clauses

The SELECT clause indicates attributes and functions are selected. The asterisk denote for all attributes of \( \text{relation.list} \).

The FROM clause indicates relations used for query.

Note: Every SQL query must have at least the SELECT clause and the FROM clause.

The WHERE clause indicates condition used to select tuples, only select tuples that satisfying the condition.

The GROUP BY clause indicates attributes, those used to group the tuples.
The HAVING clause indicates condition used to select groups, only select groups satisfying the condition.

c. The operators
The operators used to combining results of SQL queries: INTERSECT, UNION, MINUS.

3. Notation and relations used to illustrate

Notation
- \( \text{car}(\text{list of relational expressions}) \) indicates the Cartesian product of all the relational expressions.
- \( \text{attr}(\text{list of relations}) \) is the set of the attributes in the attributes schema of the specified relations.
- \( \text{attr}(\text{relational expressions}) \) is the set of attributes occurring in the results produced by the evaluation of a relational expression.
- \( \text{rels}(\text{list of attributes}) \) is the set of relations having the specified attributes.
- \( \text{rels}(\text{relational expression}) \) is the set of the relations, whose attributes appear in the relational expression.
- \( \text{extrattr}(\text{predicate}) \) is the set of attributes which appear in the predicate.
- \( \text{extrels}(\text{predicate}) \) is the set of relations whose attributes appear in the predicate.
- \( \text{meaning}(Q) \) is the relation results of query \( Q \).

The relations used to illustrate
To illustrate, we use the database relations in Date’s book [2].
- Relations \( S \) - Suppliers

<table>
<thead>
<tr>
<th>SCODE</th>
<th>SNAME</th>
<th>STATUS</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Smith</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>S2</td>
<td>Jones</td>
<td>10</td>
<td>Paris</td>
</tr>
<tr>
<td>S3</td>
<td>Blake</td>
<td>30</td>
<td>Paris</td>
</tr>
</tbody>
</table>

- Relation \( P \) - Products

<table>
<thead>
<tr>
<th>PCODE</th>
<th>PNAME</th>
<th>COLOR</th>
<th>WEIGHT</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>Nut</td>
<td>Red</td>
<td>12</td>
<td>London</td>
</tr>
<tr>
<td>P2</td>
<td>Bolt</td>
<td>Green</td>
<td>17</td>
<td>Paris</td>
</tr>
<tr>
<td>P3</td>
<td>Screw</td>
<td>Blue</td>
<td>17</td>
<td>Rome</td>
</tr>
<tr>
<td>P4</td>
<td>Screw</td>
<td>Red</td>
<td>14</td>
<td>London</td>
</tr>
</tbody>
</table>

- Relation \( SP \) - Supplier-Product

<table>
<thead>
<tr>
<th>SCODE</th>
<th>PCODE</th>
<th>QTY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>P1</td>
<td>300</td>
</tr>
<tr>
<td>S1</td>
<td>P2</td>
<td>200</td>
</tr>
<tr>
<td>S1</td>
<td>P3</td>
<td>400</td>
</tr>
<tr>
<td>S2</td>
<td>P1</td>
<td>300</td>
</tr>
<tr>
<td>S2</td>
<td>P2</td>
<td>400</td>
</tr>
<tr>
<td>S3</td>
<td>P2</td>
<td>400</td>
</tr>
</tbody>
</table>

III. SQL QUERIES EXPRESSED IN RELATIONAL CALCULUS

1. Notation and definition
Let \( Q \) be a SQL query of the form:
SELECT (selector) FROM (relation_list) WHERE (predicate)
GROUP BY (gb_attr) HAVING (hav_condition)

- Q.(component) denotes the corresponding components of Q,
- Q.(component.1).(component.2) denotes the corresponding component of Q.(component.1),
- Q.(selector).(attrs) is list of attributes in Q.(selector),
- Q.(selector).(function_list) is list of functions in Q.(selector).

In this paper we assume Q.(selector).(function_list) = \{F_i[A_j]\}, i = 1, \ldots, k, F_i[A_j] is a function \(F_i\) computed on attribute \(A_j\).

Example: Let \(R\) be relation, \(R\) has the schema \(R(A, B)\). \(Q\) be a SQL query of the form SELECT \(R.A, R.B, \text{SUM}(R.A)\) FROM \(R\), then

\[
Q.(selector) = \{R.A, R.B, \text{SUM}(R.A)\} \\
Q.(selector).(attrs) = \{R.A, R.B\} \\
Q.(selector).(function_list) = \{\text{SUM}(R.A)\} \\
\]

- \(F_i[R; U; A_j](t)\) (where \(R\) is a relation, \(U\) is a subset of attr(\(R\)), \(A_j\) is an attribute of \(R\), \(A_j \notin U\), \(t\) is a tuple in \(R\)): is the value of function \(F_i\) computed on attribute \(A_j\) of relation \(R\) with group-attribute \(U\) for tuple \(t\) of \(R\).

Note: The values of \(F_i[A_j]\) are same with tuples in a group (Fig. 1).

\[
\forall t, t' \in R, \text{if } t'(U) = t(U) \text{ then } F_i[R; U; A_j](t') = F_i[R; U; A_j](t)
\]

\[
\begin{array}{|c|c|c|}
\hline
\ldots & U & F_i[R; U; A_j] \\
\hline
r_1 & g_1 & f_1 \\
r_2 & g_1 & f_1 \\
r_3 & g_2 & f_2 \\
r_4 & g_2 & f_2 \\
r_5 & g_2 & f_2 \\
\hline
\end{array}
\]

\text{Fig. 1}

Definition 1. Let \(Q\) be a SQL query and \(E\) be a relational calculus expression, we say that \(Q\) is equivalent to \(E\) iff the results of \(Q\) and \(E\) are the same when we substitute the same relations for identical name in the two expressions.

When \(Q\) is equivalent to \(E\), we say that \(E\) is \(Q\) expressed in relational calculus and denoted \(Q = E\).

Definition 2. Let \(Q, Q'\) be SQL queries. We say that \(Q\) is equivalent to \(Q'\), denoted \(Q = Q'\), iff when we substitute the same relations for identical name in the two expressions, we get the same result.

We have \(Q = Q' \Leftrightarrow \text{meaning}(Q) = \text{meaning}(Q')\).

2. The top level query

Let \(Q\) be a SQL query of the form:

SELECT (selector) FROM (relation_list) WHERE (predicate)
GROUP BY (gb_attr) HAVING (hav_condition)

Based on the meaning of \(Q\), we have:

\[
Q = \{t(Q.(selector)) \mid \exists t'(R(t') \land t(Q.(selector).(attrs)) = t'(Q.(selector).(attrs)) \land F(t') \land R(R) \land (r(Q.(gb_attr))) = t'(Q.(gb_attr)) \rightarrow H(r))\} \land (t(F_i[A_j]) = F_i[R'; Q.(gb_attr); A_j](t') \text{ if } i = 1, \ldots, k\} \\
\]

where
\[ R = \text{car}(Q.(\text{relation_list})) \],
\[ F(t) \text{ is predicate } Q.(\text{predicate}) \],
\[ H(t) \text{ is condition } Q.(\text{hav._condition}) \],
\[ R' = \{ t \mid R(t) \land F(t) \} \text{- set of tuples of } R, \text{ those are satisfying } Q.(\text{predicate}) \],
\[ R'' = \{ t \mid R'(t) \land \forall t'(R'(t') \land (t'(Q.(\text{gb_attr})) = t(Q.(\text{gb_attr})) \rightarrow H(t'))) \} \].

We rewrite equation (1):
\[ Q = \{ t(Q.(\text{selector})) \mid \exists t'(R''(t') \land t(Q.(\text{selector}).(\text{attrs}) = t'(Q.(\text{selector}).(\text{attrs})) \land (t(F_i[A_j]) = F_i[R''; Q.(\text{gb_attr}); A_j](t') \mid i = 1, \ldots, k) \} \] (2)

3. The subquery

The subqueries may be used in conjunction with the IN, ALL, EXISTS, ... operators.

Example. Find the name of suppliers, those do not supply product P1.

\[ Q = \text{SELECT SNAME FROM S WHERE "P1" NOT IN (SELECT PCODE FROM SP WHERE SCODE=S.SCODE)} \]

We consider the subquery:

\[ Q' = \text{SELECT PCODE FROM SP WHERE SCODE=S.SCODE} \]

Remark.

1) The subquery \( Q' \) is dependent on S.SCODE of S.
2) To have meaning (Q), we need to have meaning (Q').

Definition 3 (External relation, External attribute of subquery).

Let \( Q \) be a subquery, \( R \) be a relation, if there exists attribute(s) of \( R \) appear in \( Q \), but \( R \) not in \( Q.(\text{relation_list}) \) then \( R \) is called external relation of \( Q \).

Every attribute of \( Q' \)'s external relation is called external attribute of \( Q \).

The set of all external attributes of \( Q \) is denoted by \( \text{Other}(Q) \).

Definition 4 (The relation result of subquery).

Let \( Q \) be a subquery, \( \text{Other}(Q) \) is set of all external attributes of \( Q \). \( S = \text{car}(\text{rel}(\text{other}(Q))) \).

For each \( s \in S \), \( Q(s) \) is a subquery, it is obtained by replace each attribute \( A_i \in \text{Other}(Q) \) by \( s(A_i) \).
Let \( R_s = \text{meaning}(Q(s)) \).

\[
\begin{array}{c|c|c|c|c}
S & \text{Other}(Q) & Q.(\text{selector}) & R_s & \text{Other}(Q) \\
\hline
s_1 & r_1 & & r_4 & \\
s_2 & r_2 & & r_5 & \\
s_3 & r_3 & & & \\
\end{array}
\]

\[
\begin{array}{c|c|c|c|c}
R & Q.(\text{selector}) & \text{Other}(Q) & & \\
\hline
r_1 & s_1 & & & \\
r_2 & s_1 & & & \\
r_3 & s_1 & & & \\
r_4 & s_2 & & & \\
r_5 & s_2 & & & \\
\end{array}
\]

Fig. 2. Relation result of subquery

Definition: The relation result of subquery \( Q \) is defined by expression:

\[ \{(r, s) \mid S(s) \land R_s(r) \} \] (3)
Example: Let Q be a subquery

\[ \text{SELECT PCODE FROM SP WHERE SCODE=S.SCODE} \]

\[ \text{Other(Q)} = \{ \text{SCODE, SNAME, STATUS, CITY} \} \]

\[ s_1 = (S1, \text{Smith,20,London}) \]
\[ s_2 = (S2, \text{Jones,10,Paris}) \]
\[ s_3 = (S3, \text{Blake,30,Paris}) \]

The relation result of sub query Q

<table>
<thead>
<tr>
<th>PCODE</th>
<th>SCODE</th>
<th>SNAME</th>
<th>STATUS</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>S1</td>
<td>Smith</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>P2</td>
<td>S1</td>
<td>Smith</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>P3</td>
<td>S1</td>
<td>Smith</td>
<td>20</td>
<td>London</td>
</tr>
<tr>
<td>P1</td>
<td>S2</td>
<td>Jones</td>
<td>10</td>
<td>Paris</td>
</tr>
<tr>
<td>P2</td>
<td>S2</td>
<td>Jones</td>
<td>10</td>
<td>Paris</td>
</tr>
<tr>
<td>P2</td>
<td>S3</td>
<td>Blake</td>
<td>30</td>
<td>Paris</td>
</tr>
</tbody>
</table>

By equation (1), we have:

\[ R_s = \{ t(Q.(selector)) | \exists t'(R_s'(t') \land t(Q.(selector).(attrs)) = t'(Q.(selector).(attrs)) \land (t(F_i[A_j]) = F_i[R_s'; Q.(gb_attr); A_j(t') i = 1, ..., k]) \} \]

where

\[ R_s^* = \{ r | R(r) \land F(r,s) \land \forall r'(R(r') \land F(r',s) \land (r(Q.(gb_attr)) = r'(Q.(gb_attr)) \rightarrow H(r', s))) \} \]

IV. RESULTS

Theorem 1. Let Q be a subquery of the form

\[ \text{SELECT (selector) FROM (relation_list) WHERE (predicate) GROUP BY (gb_attr) HAVING (hav_condition)} \]

then

\[ Q = \{ t(Q.(selector) \cup \text{Other(Q)}) | \exists (P_{FH}(p) \land t(Q.(selector).(attrs) \cup \text{Other(Q)}) = p(Q.(selector).(attrs) \cup \text{Other(Q)}) \land (t(F_i[A_j]) = F_i[P_{FH}; Q.(gb_attr) \cup \text{Other(Q)}; A_j(p) i = 1, ..., k]) \} \]

where

\[ P = \text{car}(Q.(relation_list) \cup \text{rels}(\text{Other(Q)})), \]
\[ P_F = \{ p | P(p) \land F(p) \}, \]
\[ P_{FH} = \{ p | P_F(p) \land \forall p'(P_F(p') \land p'(Q.(gb_attr) \cup \text{Other(Q)}) = p(Q.(gb_attr) \cup \text{Other(Q)} \rightarrow H(p')) \} \]

Define \( P^* = \{ (t,s) | S(s) \land R_s^*(t) \} \). The proof of Theorem 1 is based on Lemma 1, and Lemma 2. We omit here the proof of these lemmas.
Lemma 1. We have $P^* = P_{FH}$.

Lemma 2. For every $p \in P^*$, $p = (t, s)$, where $s \in S$, $t \in R^*_s$. Let $F_i$ be a function, $A_j \in \text{attr}(Q.(relation\_list))$ we have:

$$F_i[R^*_s; Q.(gb\_attr); A_j](t) = F_i[P^*; Q.(gb\_attr) \cup Other(Q); A_j](p) \quad (\ast)$$

Proof of Theorem 1. By equation (3) we have the equivalence of $Q$ with the expression $E$ defined by:

$$E = \{ t(Q.(selector) \cup Other(Q)) \mid \exists s (s(s) \land t(Other(Q)) = s \land \exists r (R_s(r) \land t(Q.(selector)) = r) \}$$

$$= \{ (r, s) \mid s(s) \land R_s(r) \},$$

where

$$R_s = \{ t(Q.(selector)) \mid \exists t'(R^*_s(t') \land t(Q.(selector).(attrs))) =$$

$$t'(Q.(selector).(attrs)) \land (t(F_i[A_j] = F_i[R^*_s; Q.(gb\_attr); A_j](t') \, \forall i = 1, \ldots, k)) \}.$$  

Let

$$E' = \{ t(Q.(selector) \cup Other(Q)) \mid \exists p (P_{FH}(p) \land$$

$$t(Q.(selector).(attrs) \cup Other(Q)) = p(Q.(selector).(attrs) \cup Other(Q)) \land$$

$$(t(F_i[A_j] = F_i[P^*; Q.(gb\_attr) \cup Other(Q); A_j](p) \, \forall i = 1, \ldots, k)) \}.$$  

By Lemma 1, we have

$$E' = \{ t(Q.(selector) \cup Other(Q)) \mid \exists p (P^*(p) \land$$

$$t(Q.(selector).(attrs) \cup Other(Q)) = p(Q.(selector).(attrs) \cup Other(Q)) \land$$

$$(t(F_i[A_j] = F_i[P^*; Q.(gb\_attr) \cup Other(Q); A_j](p) \, \forall i = 1, \ldots, k)) \}.$$  

We show that $E = E'$

a) First we show that $E' \subseteq E$

$$\forall t \in E' \Rightarrow \exists p \in P^*$$

$$t(Q.(selector).(attrs) \cup Other(Q)) = p(Q.(selector).(attrs) \cup Other(Q))$$

$$t(F_i[A_j] = F_i[P^*; Q.(gb\_attr) \cup Other(Q); A_j](p) \, \forall i = 1, \ldots, k.$$  

Since $p \in P^*$ then $p = (r, s)$ where $s \in S, r \in R^*_s$.

We have $t(Other(Q)) = p(Other(Q)) = s \in S$

$$t(Q.(selector).(attrs) = p(Q.(selector).(attrs)) = r(Q.(selector).(attrs))$$

$$t(F_i[A_j] = F_i[P^*; Q.(gb\_attr) \cup Other(Q); A_j](p) \, \forall i = 1, \ldots, k.$$  

By Lemma 2, we have

$$t(F_i[A_j] = F_i[R^*_s; Q.(gb\_attr); A_j](t) \, \forall i = 1, \ldots, k.$$  

Let $t' = t(Q.(selector))$ then

$$t'(Q.(selector).(attrs)) = t'(Q.(selector).(attrs)) = r(Q.(selector).(attrs))$$

and $t'(F_i[A_j] = F_i[P^*; Q.(gb\_attr) \cup Other(Q); A_j](p) = F_i[R^*_s; Q.(gb\_attr); A_j](r) \, \forall i = 1, \ldots, k$ so $t' \in R^*_s$.

Clearly $t = (t', s) \Rightarrow t \in E$ so $E' \subseteq E \quad (\ast)$

b) Now we have to show that $E \subseteq E'$

$$\forall t \in E \Rightarrow t = (r, s) \text{ where } s \in S, r \in R^*_s.$$
SQL QUERY EXPRESSED IN RELATIONAL CALCULUS

Since \( r \in R_* \) then \( \exists t' \in R_*^t, r(Q.(selector).(attrs)) = t'(Q.(selector).(attrs)) \) and \( r(F_i[A_j]) = F_i[R_*^t; Q.(gb\_attr); A_j](t') \forall i = 1, \ldots, k \).

Let \( p = (t', s) \), we have \( p \in P^* \) and
\[
t(Q.(selector).(attrs)) = r(Q.(selector).(attrs)) = t'(Q.(selector).(attrs))
= p(Q.(selector).(attrs))
\]
\[
t(F_i[A_j]) = F_i[R_*^t; Q.(gb\_attr); A_j](t') \quad \forall i = 1, \ldots, k
\]
By Lemma 2, we have
\[
t(F_i[A_j]) = F_i[P^*; Q.(gb\_attr) \cup Other(Q); A_j](p) \forall i = 1, \ldots, k \quad \text{so} \quad t \in E' \Rightarrow E' \subseteq E \quad (**)
\]
From (*) and (**) \( \Rightarrow E = E' \). \( \Box \)

By Theorem 1, we have the following remark:

**Remark 1.** Let \( Q \) be a query of the form

\[\text{SELECT } \langle \text{selector} \rangle \text{ FROM } \langle \text{relation\_list} \rangle \text{ WHERE } \langle \text{predicate} \rangle \text{ GROUP BY } \langle \text{gb\_attr} \rangle \text{ HAVING } \langle \text{hav\_condition} \rangle\]

then \( Q \) is equivalent to the query \( Q' \) of the form

\[\text{SELECT } \langle \text{selector} \rangle \cup Other(Q) \text{ FROM } \langle \text{relation\_list} \rangle \cup \text{rels}(Other(Q)) \text{ WHERE } \langle \text{predicate} \rangle \text{ GROUP BY } \langle \text{gb\_attr} \rangle \cup Other(Q) \text{ HAVING } \langle \text{hav\_condition} \rangle.\]

**Theorem 2.** Let \( Q \) be a query of the form

\[\text{SELECT } \langle \text{selector} \rangle \text{ FROM } \langle \text{relation\_list} \rangle \text{ WHERE } \langle \text{predicate} \rangle \text{ GROUP BY } \langle \text{gb\_attr} \rangle \text{ HAVING } \langle \text{hav\_condition} \rangle\]

then we have
\[
Q = \{t(Q.(selector) \cup Other(Q)) | \exists r(R^*(r) \land t(Q.(selector).(attrs) \cup Other(Q)) =
\]
\[
r(Q.(selector).(attrs) \cup Other(Q)) \land
\]
\[
(t(F_i[A_j]) = F_i[R^*; Q.(gb\_attr) \cup Other(Q); A_j](r) \forall i = 1, \ldots, k)\}
\]

where \( Q.(ngb\_query)* \) denotes the query of the form

\[\text{SELECT } * \text{ FROM } Q.(relation\_list) \text{ WHERE } Q.(predicate)\]

\[R = \text{car}(\text{meaning}(Q.(ngb\_query)*)) \cup \text{rels}(Other(Q)) - \text{rels}(\text{meaning}(Q.(ngb\_query)*))\]

\[R^* = \{r \mid R(r) \land \forall r'(R'(r') \land (r'(Q.(gb\_attr) \cup Other(Q)) = r(Q.(gb\_attr) \cup Other(Q)) \rightarrow H'(r'))\}\].

Theorem 2 allows to express queries with GROUP BY clause by the result of queries without GROUP BY clause.

The proof of Theorem 2 is based on the Lemma 3. As above, we omit here the proof of Lemma 3.

**Lemma 3.** We have \( R = \{p | P(p) \land F(p)\} \).

**Proof of Theorem 2.**

By Lemma 3 we have \( R^* = P_{FH} \).

By Theorem 1 clearly
\[
Q = \{t(Q.(selector) \cup Other(Q)) | \exists r(R^*(r) \land
\]
\[
t(Q.(selector).(attrs) \cup Other(Q)) = r(Q.(selector).(attrs) \cup Other(Q)) \land
\]
\[
(t(F_i[A_j]) = F_i[R^*; Q.(gb\_attr) \cup Other(Q); A_j](r) \forall i = 1, \ldots, k)\}\]. \( \Box \)

By Theorem 2, we have the following remark:
Remark 2. Let $Q$ be a query of the form

$$\text{SELECT } (\text{selector}) \text{ FROM } (\text{relation\_list}) \text{ WHERE } (\text{predicate}) \text{ GROUP BY } (\text{gb\_attr}) \text{ HAVING } (\text{hav\_condition})$$

then the following queries are equivalent:

i) $Q$

ii) $Q' = \text{SELECT } Q.(\text{selector}) \cup \text{Other}(Q) \text{ FROM } \text{TEMP1} \text{ GROUP BY } Q.(\text{gb\_attr}) \cup \text{Other}(Q)$, where $\text{TEMP1} = \text{meaning}(Q^*)$

with $Q^* = \text{SELECT } * \text{ FROM } Q.(\text{relation\_list}) \text{ WHERE } Q.(\text{predicate}) \text{ GROUP BY } Q.(\text{gb\_attr}) \text{ HAVING } Q.(\text{hav\_condition})$.

iii) $Q'' = \text{SELECT } Q.(\text{selector}) \cup \text{Other}(Q.(\text{ngb\_query})) \text{ FROM } \text{TEMP2} \text{ GROUP BY } Q.(\text{gb\_attr}) \cup \text{Other}(Q.(\text{ngb\_query})) \text{ HAVING } Q.(\text{hav\_condition})$, where $\text{TEMP2} = \text{meaning}(Q.(\text{ngb\_query})^*)$.

REFERENCES


Tóm tắt. Trong bài báo này chúng tôi trình bày một số kết quả về sự tương đương giữa những câu hỏi của SQL và biểu thức trong phép tính quan hệ, một số tính chất của câu hỏi trong SQL. Những kết quả này được sử dụng cho việc chuyển dịch câu hỏi của SQL vào đại số quan hệ.

(1) Institute of Information Technology.
(2) Pedagogical Institute of Quy Nhơn.

Received: March 11, 1998