DIALOGUE ORGANIZATION WITH USERS IN INFORMATION SYSTEMS OF PRODUCTION MANAGEMENT

V.G. Mokrozub, Yu.V. Nemtinova, S.V. Morozov

Department "Automated Designing of Technological Equipment", TSTU; mokr@mail.gaps.tstu.ru

Represented by a Member of the Editorial Board Professor V.I. Konovalov

Key words and phrases: information system; relational database; user’s query; output format.

Abstract: The article considers dialogue organization between users and information systems based on relational databases, using programming language understandable to users. It also describes the means of conversion of a query line to operators of SQL language.

Introduction

Modern management systems contain great volumes of information which is processed and provided in certain format on users’ queries. At the process of information systems design it is impossible to predict all possible combinations of future queries. Hence, an information system should allow users to make queries using understandable language and to provide results in a format defined by user.

Nowadays the most often used model of information systems is the relational model. The basic programming language of modern relational databases (RDB) is Structured Query Language (SQL).

The purpose of this article is developing of simple queries syntax and their conversion to SQL syntax. The publications on this topic mainly describe what has to be done to formulate users’ queries using understandable language and to lesser extend what has to be done to minimize programming burden [1, 2].

For the purpose of our discussion we divide RDB into two classes depending on their structure:

– traditional databases or databases with flat tables;
– databases of OBJECT–ATTRIBUTE–VALUE type.

By traditional RDB we will indicate databases where entities are presented by wide tables. For example, the information entity "Product delivery to a warehouse by production unit" is characterized by attributes: "Product", "Quantity", "Production_Unit", "Date_of_manufacturing", "Delivery_date", etc. Here RDB is represented by relation (table) where all listed attributes (fields) are presented.

Databases of OBJECT–ATTRIBUTE–VALUE type are represented by three basic tables: "Register of objects", "Register of attributes", "Values of objects’ attributes". Last table has connections with the first two of “many to one” type.
1. Dialogue Organization in Traditional Databases

Let’s present normalized RDB $BD_T$ as follows:

$$BD_T = \{R_1, ..., R_i, ..., R_I\}, \quad i = \overline{1, I};$$

$$R_i = \{PK_i, FK_i, D_i\} ;$$

$$FK_i = \{FK_{i1}, ..., FK_{ij}, ..., FK_{iJ}\}, \quad j = \overline{1, J};$$

$$D_j = \{D_{j1}, ..., D_{jk}, ..., D_{jK}\}, \quad k = \overline{1, K};$$

where $R_i$ – $i$-th relation (table); $I$ – number of relations in a database; $PK_i$ – key attribute of $i$-th relation; $FK_i$ – set of attributes for external keys of $i$-th relation; $J$ – number of attributes for external keys; $D_i$ – set of data attributes of $i$-th relation; $K$ – number of data attributes (indices $i$, $J$ and $K$ are set aside).

Let’s put $W = \{W_1, ..., W_i, ..., W_T\}, \quad t = \overline{1, T}, \quad W_i \in \bigcup_i D_i$ – set of attributes, which can be used to make a query or for data output. By elementary condition of a query $p$ we will understand the notion $p = <a, f, z>$, where $a$ – attribute $a \in W \quad z$ – set of attribute values, $f$ – condition (equal, not equal, greater, lesser, etc.), for example $< \text{City} = 'Moscow'>$ or $< \text{Data} = '01.01.2008'>$.

User’s inquiry $S$ is represented by a set of elementary queries, which are connected by logic operators AND, OR, NOT:

$$S = F(p_1, ..., p_m, ..., p_M), \quad m = \overline{1, M};$$

where $F$ – function which connects elementary conditions of queries; $M$ – number of elementary conditions; for example $S = p_1 \quad \text{AND}(p_2 \quad \text{OR} \quad p_3) \quad \text{AND}(\text{NOT} \quad p_4)$.

The output of a query is composed by elements of set $B$, which consists of elements of set $W$:

$$B = \{B^1, B^2\};$$

$$B^1 = \{b_1^1, b_{c1}^1, ..., b_c^1\}, \quad b_c^1 \in W, \quad c = \overline{1, C};$$

$$B^2 = \{\varphi(b_l^2), ..., \varphi_l^2(b_L^2), ..., \varphi_L(b_L^2)\}, \quad b_l^2 \not\in W, \quad b_l^2 \not\in B^1, \quad l = \overline{1, L};$$

where $B^1$ – set of data attributes, which are components of output, and where such aggregate functions as (Sum, Max and others) are not applicable; $b_l^2, \quad l = \overline{1, L}$ – set of data attributes, which are components of output, and where aggregate functions are applicable; $\varphi_l, \quad l = \overline{1, L}$ – aggregate functions, which are applicable to elements $b_l^2$. For example $B^1 = \{\text{Product}, \text{City}\}$, $B^2 = \{\text{Sum(Amount)}\}$.

The process of making a query and obtain its results is divided into the following steps:

– input of elementary conditions, $\{p_1, ..., p_o, ..., p_O\}$;

– input of a query line, $S = F(p_1, ..., p_m, ..., p_M), \quad m = \overline{1, M}$;
– input of output fields \( B^1 = \{b^1_1, b^1_2, ..., b^1_{cL}\} \), \( b^1_i \in W \), \( c = 1, C \);
– input of output fields \( B^2 = \{\phi_1(b^2_1), ..., \phi_l(b^2_l), ..., \phi_L(b^2_L)\} \), \( b^2_l \in W \), \( b^2_l \not\in B^1 \), \( l = 1, L \);
– execution of a query and obtaining an output.

To convert a query line \( S \) into SQL format elementary queries formulated as \( < a, f, z > \), are represented as \( < aID, f, zID > \), where \( aID \) – name of primary key for \( a \); \( zID \) – set of primary key values field for set of attributes \( z \). For example, elementary condition \( < \text{City} = 'Moscow' > \) in programming has to be changed for \( < \text{IDCity} = \text{IDMoscow} > \), where \( \text{IDMoscow} \) – value of key field of value 'Moscow'. At the same time elementary condition \( < \text{Data} \geq 01.01.2008 > \) remains unchanged. In the first case we name the elementary condition as ‘condition for ID’, in the second case as ‘condition for value’. Converted query line \( S \) is named \( SI \).

In order to organize the dialogue with users at elementary conditions input it is necessary to provide the choice of attribute \( a \) and input (choice) of attribute values \( z \). For \( a \) we create an attribute table \( \{AID, AIDG\} \). Values (domains) of the field \( A \) are given by set \( W = \{W_1, ..., W_l, ..., W_T\} \). Domain of the field \( AID \) are names of key fields for \( W = \{W_1, ..., W_l, ..., W_T\} \), if it is an elementary condition for \( ID \) or names if it is an elementary conditions for value. Elementary conditions are kept in the table \( Y\{\text{Condition\_name, AID, f, Z, zID}\} \).

Let’s put \( \{PK_1, ..., PK_l, ..., PK_R, W_1, ..., W_l, ..., W_T\} \) – representation (View), by which output is composed. Then a query on SQL can be written as:

\[
\text{select } B \text{ from } V \text{ where } SI \text{ group by } B^1 \text{ order by } B^1.
\]

Let’s examine the program dialogue on the example of products’ delivery to consumers. Each product is produced by defined production unit. The database is represented as

\[
BDT = \{R_1, R_2, R_3, R_4, R_5\},
\]

where \( R_1 \) (IDR1, IDConsumer, IDProduct, Quantity, Delivery_date) – delivery of a product; \( R_2 \) (IDProduct, IDProduction_unit, Product_title) – products; \( R_3 \) (IDConsumer, IDCity, Consumer_title) – consumers; \( R_4 \) (IDCity, City_title) – cities; \( R_5 \) (ID_Production_unit, Production_unit_title) – production floors.

Thus, delivery is characterized by six attributes "Consumer", "Product", "Quantity", "Delivery date", "Production unit", "City".

User can formulate a query to get an output with any combination of attributes, for example, delivery report for products manufactured by unit 1, 3, 10 in Lipetsk, Tambow in January 2008 and January 2009. Forms of output can be various, for example, Product–Quantity, City–Product–Quantity, Product–City–Quantity, etc.

Set \( W = \{\text{Consumer\_title, Product\_title, City\_title, Production\_unit\_title, Delivery\_date, Quantity}\} \).

For the given example attribute table of queries \( G \) is presented as (Tab. 1).

To get an output \( \{PK_1, ..., PK_l, ..., PK_R, W_1, ..., W_l, ..., W_T\} = \{\text{IDR1, IDConsumer, Consumer\_title, IDProduct, Product\_title, IDCity, City\_title, ID_Production\_unit, Production\_unit\_Delivery\_date, Quantity}\} \) the query has to be as follows:

290 ISSN 0136-5835. Вестник ТГТУ. 2010. Том 16. № 2. Transactions TSTU.
Table 1

Query attributes $G$

<table>
<thead>
<tr>
<th>IDG</th>
<th>$A$</th>
<th>AID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Consumer_title</td>
<td>IDConsumer</td>
</tr>
<tr>
<td>2</td>
<td>Product_title</td>
<td>IDProduct</td>
</tr>
<tr>
<td>3</td>
<td>Production_unit_title</td>
<td>IDProduction_unit</td>
</tr>
<tr>
<td>4</td>
<td>City_title</td>
<td>IDCity</td>
</tr>
<tr>
<td>5</td>
<td>Delivery_date</td>
<td>Delivery_date</td>
</tr>
<tr>
<td>6</td>
<td>Quantity</td>
<td>Quantity</td>
</tr>
</tbody>
</table>

select IDR1, IDConsumer, Consumer_title, IDProduct, Product_title, IDCity, City_title, ID_Production_unit, Production_unit_title, Delivery_date, Quantity
from $R_1, R_2, R_3, R_4, R_5$
where $R_1$.IDConsumer = $R_3$.IDConsumer
and $R_1$.IDProduct = $R_2$.IDProduct
and $R_2$.IDProduction_unit = $R_5$.IDProduction_unit
and $R_3$.IDCity = $R_4$.IDCity.

Query line is represented by logic formulation of elementary conditions. For example, query line “delivery of products manufactured by unit 1, 3, 10, (IDProduction_unit - 2,5,8 respectively) in Moscow, Lipetsk, Tambov (IDCity – 10,11,12 respectively), in January 2008 and January 2009” will be represented as follows:

$$S = F(p_1, p_2, p_3, p_4, p_5, p_6) = p_1 \text{ AND } p_2 \text{ AND } (p_3 \text{ AND } p_4 \text{ OR } p_5 \text{ AND } p_6),$$

where $p_1$ – production unit = (production unit 1, production unit 3, production unit 10); $p_2$ – City = (Moscow, Lipetsk, Tambov); $p_3$ – Delivery_date ≥ 01.01.2008; $p_4$ – Delivery_date ≤ 31.01.2008; $p_5$ – Delivery_date ≥ 01.01.2009; $p_6$ – Delivery_date ≤ 31.01.2009.

Input of every elementary condition $< a, f, z >$ consists of the following steps:
- attribute choice $a$ from the table of query attributes $G$;
- set the condition, $f$;
- input of value or set of values, $z$.
Together with this we create Tab. 2, columns 1, 2, 4, 5 which are presented to user.

Table 2

<table>
<thead>
<tr>
<th>Condition</th>
<th>$A$</th>
<th>AID</th>
<th>$f$</th>
<th>$Z$</th>
<th>ZID</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_1$</td>
<td>Production_floor_title</td>
<td>IDProduction_unit</td>
<td>= production unit 1, production floor 3, production floor 10</td>
<td>2, 5, 8</td>
<td></td>
</tr>
<tr>
<td>$p_2$</td>
<td>City_title</td>
<td>IDCity</td>
<td>= Moscow, Lipetsk, Tambov</td>
<td>10, 11, 12</td>
<td></td>
</tr>
<tr>
<td>$p_3$</td>
<td>Delivery_date</td>
<td>Delivery_date</td>
<td>≥ 01.01.2008</td>
<td>01.01.2008</td>
<td></td>
</tr>
<tr>
<td>$p_4$</td>
<td>Delivery_date</td>
<td>Delivery_date</td>
<td>≤ 31.01.2008</td>
<td>31.01.2008</td>
<td></td>
</tr>
<tr>
<td>$p_5$</td>
<td>Delivery_date</td>
<td>Delivery_date</td>
<td>≥ 01.01.2009</td>
<td>01.01.2009</td>
<td></td>
</tr>
<tr>
<td>$p_6$</td>
<td>Delivery_date</td>
<td>Delivery_date</td>
<td>≤ 31.01.2009</td>
<td>31.01.2009</td>
<td></td>
</tr>
</tbody>
</table>
Than user inputs condition query line \( S = p_1 \text{AND} p_2 \text{AND} (p_3 \text{AND} p_4 \text{OR} p_5 \text{AND} p_6) \). Undoubtedly users have to be able to compose logical expressions. Practical experience shows that half hour training allows any user to get acquainted with this aspect.

Output fields (fields of \( V \) representation), are also chosen by users from a table of attributes \( G \). As a result we obtain the following sets: \( B = \{ \text{Product_title, Production_unit_title, sum (Quantity)} \} \), \( B^1 = \{ \text{Product_title, Production_unit_title} \} \), \( B^2 = \{ \text{sum(Quantity)} \} \).

Query condition line \( S \) based on table \( Y \) (Table 2) is converted into line \( SI : \)

\[
SI = p_1 \text{AND} p_2 \text{AND} ( p_3 \text{AND} p_4 \text{OR} p_5 \text{AND} p_6 ),
\]

where \( p_1 \) – IDproduction_unit in \( z_1 \); \( p_2 \) – IDCity in \( z_2 \); \( p_3 \) – Delivery_date \( \geq \) \( z_3 \); \( p_4 \) – Delivery_date \( \leq \) \( z_4 \); \( p_5 \) – Delivery_date \( \geq \) \( z_5 \); \( p_6 \) – Delivery_date \( \leq \) \( z_6 \); \( z_1 = (2, 5, 8); z_2 = (10, 11, 12); z_3 = 01.01.2008; z_4 = 31.01.2008; z_5 = 01.01.2009; z_6 = 31.01.2009. As a result we formulate a query:

\[
\text{select } B \text{ from } V \text{ where } SI \text{ group by } B^1 \text{ order by } B^1 .
\]

2. Dialogue Organization in Databases of Object–Attribute–Attribute Value Type

Examples of databases with this type of structure are database of equipment, reference of substances’ properties, etc.

The essence of database query is searching of such objects whose attributes satisfy to required conditions. For example, making a search of capacitive apparatus which have anchor-impeller mixer and volume more than 10 cubic meters in the database of equipment. Database \( BDO \) in this case can be presented as

\[
BDO = \{ O, S, SO, SZ \},
\]

where \( O(\text{IDO, Object_title}) \) – objects; \( SW(\text{IDSW, Attribute_title}) \) – attributes; \( SO(\text{IDSO, IDO, IDSW, Attribute_value, IDSZ}) \) – attribute values of the concrete objects; \( SZ(\text{IDSZ,IDSW, Probable_value}) \) – list of attribute values.

Relation \( SZ \) is entered for those attributes, whose values are chosen from a concrete list, for example mixer type is selected form the list: blade, turbine, frame, etc.

Table of attributes is presented in explicit form (\( SW \)) and it is not necessary to create it as described above. Also there is no need to create representation \( V \), as its role is executed by tables \( O \) and \( SO \).

The sequence of making a query and its proceeding will be examined on the following example. Make a search of capacitive apparatus which have turbine mixer and volume less than 10 m\(^3\) in the base of equipment.

Query \( S \) will consist of two elementary condition connected by logical \( \text{AND} \):

\[
"\text{volume of apparatus is less than 10 m}^3\text{." }\text{"mixer type – turbine".}
\]

Input of each elementary condition \( < a, f, z > \) consist of the following steps:

– attribute choice \( a \) from attribute table \( SW \);
– set of conditions, \( f \);
– input of value \( z \) or their choice from the table \( SZ \).

Together with this relation \( Y \) (Tab. 3) is created, and its columns 1, 2, 4, 5 are presented to users.
Table 3

<table>
<thead>
<tr>
<th>Condition name</th>
<th>( A ) (S.Attribute_name)</th>
<th>( AID ) (S.IDS)</th>
<th>( f ) (SZ.Possible_value or constant)</th>
<th>( Z ) (SZ.IDS or Null)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p_1 )</td>
<td>Mixer type</td>
<td>5</td>
<td>=</td>
<td>Turbine</td>
</tr>
<tr>
<td>( p_2 )</td>
<td>Device volume, cubic m.</td>
<td>7</td>
<td>&lt;</td>
<td>10</td>
</tr>
</tbody>
</table>

Than user formulates a query

\[ S = p_1 \text{ AND } p_2, \]

where \( p_1 \) – Mixer.type = turbine; \( p_2 \) – Device volume < 10.

Query condition line based on table \( Y \) (See Tab. 3) is converted into line:

\[ SI = p_{i1} \text{ AND } p_{i2}, \]

where \( p_{i1} \) – exists (select * from SO where O.IDO = SO.IDO and SO.IDS = 5 and SO.IDSZ = 15); \( p_{i2} \) – exists (select * from SO where O.IDO = SO.IDO and SO.IDS = 7 and SO.Attribute_value < 10).

It is not complicated to create elementary conditions \( p_i \) as they have similar part "exists (select * from SO where O.IDO = SO.IDO and" and variable part, which is composed based on Tab. 3.

Thus a query on object search, which satisfy conditions \( SI \), can be written in the form:

\[ \text{select * from } O \text{ where } SI \]
Fig. 1 shows a sequence of dialogue organization in the information system of repair management system of public corporation "Pigment" [3], which contain equipment database. Numbers above the arrows 1, 2, ..., 8 signifies sequence of user’s actions.

Conclusion

The described above means of query setup was used by the authors in the frame of corporate information system of public company "Pigment" at sales department for delivery information search and at the plant engineer department for search of equipment with given characteristics. Three years use of this approach has proved its effectiveness. This approach is of universal use and can be implemented in information systems of various purposes.

References


3. Мокрозуб, В.Г. Процедурные и информационно-логические модели планирования выпуска продукции и ремонтов технологического оборудования многоассортиментных производств / В.Г. Мокрозуб, С.Я. Егоров, В.А. Немтинов // Информац. технологии в проектировании и производстве. – 2009. – № 2. – С. 72–76.

Организация диалога с пользователями в информационных системах менеджмента предприятия

В.Г. Мокрозуб, Ю.В. Немтинова, С.В. Морозов

Кафедра «Автоматизированное проектирование технологического оборудования», ГОУ ВПО «ТГТУ»; mokr@mail.gaps.tstu.ru

Ключевые слова и фразы: запрос пользователя; информационная система; реляционная база данных; формат вывода.

Аннотация: Рассматривается организация диалога пользователей с информационными системами, основанными на реляционных базах данных, на языке, понятном пользователю. Описаны способы преобразования строки запроса пользователя в операторы языка SQL.
Organisation des Dialoges mit den Nutzern
in den Informationssystemen des Betriebsmanagements

Zusammenfassung: Es wird die Organisation des Dialoges der Nutzer mit den Informationssystemen, die sich auf die Relationsdatenbanken stützen, betrachtet. Es sind die Verfahren der Transformation der Zeile der Nutzersanfrage in die Operatoren der SQL-Sprache beschrieben.

Organisation du dialogue avec les usagers dans les systèmes informatiques du management de l’entreprise

Résumé: Est examinée l’organisation du dialogue des usagers avec les systèmes informatiques fondés sur les bases de relation sur la langue compréhensible à l’usager. Sont décrits les moyens de la transformation de la ligne de la demande de l’usager dans l’opérateur de la langue SQL.


Рецензент: Немтинов Владимир Алексеевич – доктор технических наук, профессор, заведующий кафедрой «Автоматизированное проектирование технологического оборудования», ГОУ ВПО «ТГТУ».