УДК 004.4

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 BDT as follows:

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

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

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

$$D_i = \{D_{i1},...,D_{ik},...D_{iK}\},\ k = \overline{1,K};$$

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).

Lets put $W = \{W_1,...,W_t,...,W_T\}$, $t = \overline{1,T}$, $W_t \in \bigcup_i^I D_i$ – 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 $a \in W$, z – set of attribute values, f – condition (equal, not equal, greater, lesser, etc.), for example < City = = 'Moscow' > or < 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:

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

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

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

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

$$B^1 = \{b_1^1,...b_c^1,...,b_C^1\},\ b^1{}_c \in W,\ \ c = \overline{1,C};$$

$$B^2 = \{\varphi_1(b_1^2),...,\varphi_l(b_l^2),...,\varphi_L(b_L^2)\},\ \ b_l^2 \notin W,\ \ b_l^2 \notin B^1,\ \ l = \overline{1,L};$$

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, $l = \overline{1,L}$ – set of data attributes, which are components of output, and where aggregate functions are applicable; φ_l, $l = \overline{1,L}$ – aggregate functions, which are applicable to elements b_l^2. For example $B^1 = \{\text{Product, City}\}$, $B^2 = \{\text{Sum(Amount)}\}$.

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

– input of output fields $B^1 = \{b_1^1,...b_c^1,...,b_C^1\}$, $b_c^1 \in W$, $c = \overline{1,C}$;

– input of output fields $B^2 = \{\varphi_1(b_1^2),...,\varphi_l(b_l^2),...,\varphi_L(b_L^2)\}$, $b_l^2 \in W$, $b_l^2 \notin B^1$, $l = \overline{1,L}$;

– 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 < City = 'Moscow' > in programming has to be changed for < IDCity = = IDMoscow >, where IDMoscow – value of key field of value 'Moscow'. At the same time elementary condition < Data ≥ 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 $G = \{GID, A, AID\}$. Values (domains) of the field A are given bye set $W = \{W_1,...,W_t,...,W_T\}$. Domain of the field AID are names of key fields for $W = \{W_1,...,W_t,...,W_T\}$, 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 $Y\{Condition\_name, A, AID, f, Z, ZID\}$.

Let's put $V\{PK_1,...,PK_i,...,PK_n,W_1,...,W_t,...,W_T\}$ – representation (View), by which output is composed. Then a query on SQL can be written as:

select B from V where SI group by B^1 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 Moscow, 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 ={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 $V\{PK_1,...,PK_i,...,PK_n,W_1,...,W_t,...,W_T\}$ = V\{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:

Table 1

Query attributes *G*

IDG	*A*	*AID*
1	Consumer_title	IDConsumer
2	Product_title	IDProduct
3	Production_unit_title	IDProduction unit
4	City_title	IDCity
5	Delivery_date	Delivery_date
6	Quantity	Quantity

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_unitr = R_5 .IDProduction_unit
and R_3 .IDCtiy = 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:

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

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 (Relation) *Y*

Condition	*A*	*AID*	*f*	*Z*	*ZID*
p_1	Production_floor_title	IDProduction_unit	=	production unit 1, production floor 3, production floor 10	2, 5, 8
p_2	City_title	IDCtiy	=	Moscow, Lipetsk, Tambov	10, 11, 12
p_3	Delivery_date	Delivery_date	≥	01.01.2008	01.01.2008
p_4	Delivery_date	Delivery_date	≤	31.01.2008	31.01.2008
p_5	Delivery_date	Delivery_date	≥	01.01.2009	01.01.2009
p_6	Delivery_date	Delivery_date	≤	31.01.2009	31.01.2009

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: $B = \{\text{Product\_title, Production\_unit\_title, sum (Quantity)}\}$, $B^1 = \{\text{Ptoduct\_title, Production\_unit\_title}\}$, $B^2 = \{\text{sum(Quantity)}\}$.

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

$$SI = pi_1 \text{ AND } pi_2 \text{ AND } (pi_3 \text{ AND } pi_4 \text{ OR } pi_5 \text{ AND } pi_6),$$

where pi_1 – IDproduction_unitr *in* zi_1 ; pi_2 – IDCity *in* zi_2 ; pi_3 – Delivery_date $\geq$ $\geq zi_3$; pi_4 – Delivery_date $\leq zi_4$; pi_5 – Delivery_date $\geq zi_5$; pi_6 – Delivery_date $\leq$ $\leq zi_6$; zi_1 = (2, 5, 8); zi_2 = (10, 11, 12); zi_3 = 01.01.2008; zi_4 = 31.01.2008; zi_5 = 01.01.2009; zi_6 = 31.01.2009.

As a result we formulate a query*:*

select B from V where SI group by B^1 order by B^1 .

2. Dialogue Organization in Databases of Object–Attribute–Attribuye 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(IDO, Object_title) – objects; SW (IDSW, Attribute_title) – attributes; SO (IDSO, IDO, IDSW, Attribute_value, IDSZ) – attribute values of the concrete objects; SZ (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 AND: "volume of apparatus is less than 10 m^3." "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 (Relation) *Y*

Condition name	*A* (S.Attribute_name)	*AID* (S.IDS)	*f*	*Z* (SZ.Possible_value or constant)	*ZID* (SZ.IDS or Null)
1	2	3	4	5	6
p_1	Mixer type	5	=	Turbine	15
p_2	Device volume, cubic m.	7	<	10	Null

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:

$$SI = pi_1 \text{ AND } pi_2 ,$$

where pi_1 – exists *(*select * from SO where O.IDO = SO.IDO and SO.IDS = 5 and SO.IDSZ = 15); pi_2 – 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 *pi* 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:

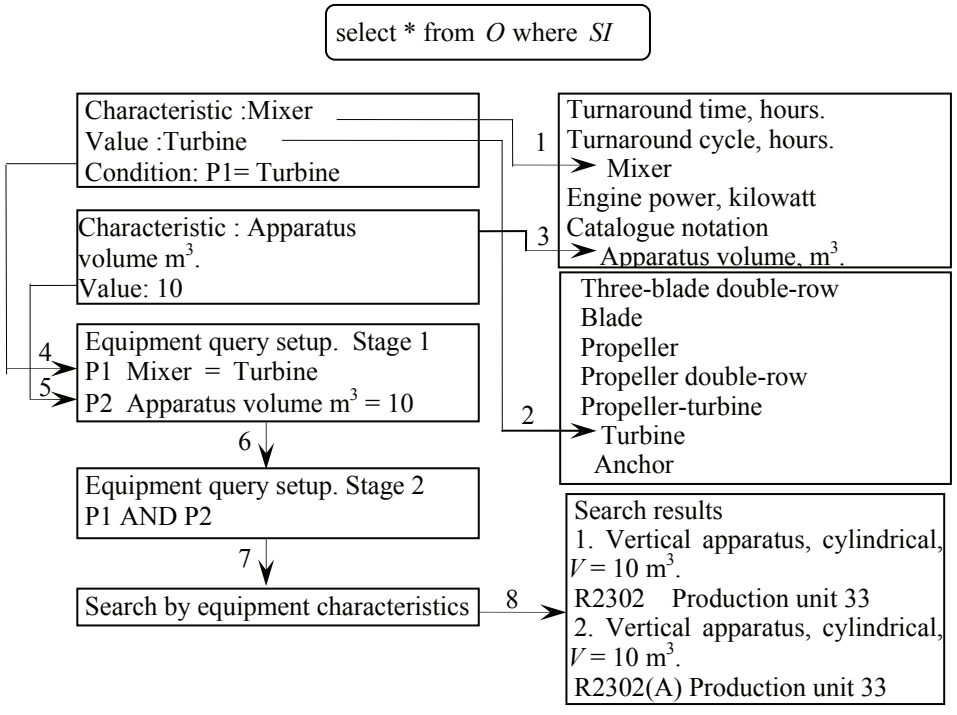

Fig. 1. Dialogue organization for equipment query setup

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.

Работа выполнена в рамках проекта НК-421(2) Федеральной целевой программы «Научные и научно-педагогические кадры инновационной России на 2009–2013 годы», направление «Информатика».

References

1. Биряльцев, Е.В. Онтологии реляционных баз данных. Лингвистический аспект [Электронный ресурс] / Е.В. Биряльцев, А.М. Гусенков // Междунар. конф. по компьютерной лингвистике : тр. междун. конф. «Диалог 2007». – Режим доступа : http://www.dialog-21.ru/dialog2007/materials/html/08.htm. – Загл. с экрана.

2. Курбатов, С.С. Автоматизированное построение естественно-языкового интерфейса для реляционных баз данных / С.С. Курбатов // Новости искусств. интеллекта. – 2002. – № 2. – С. 17–21.

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 Informationssytemen des Betriebsmanagementes

Zusammenfassung: Es wird die Organisation des Dialoges der Nutzer mit den Informationssystemen, die sich auf die Relationsdatenbänke 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.

Авторы: ***Мокрозуб Владимир Григорьевич*** – кандидат технических наук, доцент, профессор кафедры «Автоматизированное проектирование технологического оборудования»; ***Немтинова Юлия Владимировна*** – кандидат экономических наук, старший преподаватель кафедры «Автоматизированное проектирование технологического оборудования»; ***Морозов Сергей Владимирович*** – магистрант кафедры «Автоматизированное проектирование технологического оборудования», ГОУ ВПО «ТГТУ».

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