Internal distribution code:

(A) [-] Publication in OJ
(B) [-] To Chairmen and Members
(C) [-] To Chairmen
(D) [X] No distribution

Datasheet for the decision of 8 September 2016

Case Number:	T 0104/12 - 3.5.07
Application Number:	04740649.1
Publication Number:	1654677
IPC:	G06F17/30
Language of the proceedings:	EN

Title of invention:
Metadata based query

Applicant:
SAP SE

Headword:
Metadata-based query/SAP

Relevant legal provisions:
EPC Art. 56

Keyword:
Inventive step - all requests (no)

Decisions cited:

Catchword:

Case Number: T 0104/12 - 3.5.07

D E C I S I O N
# of Technical Board of Appeal 3.5.07
of 8 September 2016

Appellant: (Applicant)	SAP SE Dietmar-Hopp-Allee 16 69190 Walldorf (DE)
Representative:	Richardt Patentanwälte PartG mbB Wilhelmstraße 7 65185 Wiesbaden (DE)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 21 September 2011 refusing European patent application No. 04740649.1 pursuant to Article 97(2) EPC.**

Summary of Facts and Submissions

I. The applicant (appellant) appealed against the decision of the Examining Division refusing European patent application No. 04740649.1.

II. The documents cited in the decision under appeal included the following:

D1: "Oracle Warehouse Builder: A Technical Overview", Oracle Technical White Paper, February 2000; and

D3: Agrawal et al.: "Storage and Querying of E-Commerce Data", Proceedings of the 27th International Conference on Very Large Data Bases, pages 149-158, September 2001.

The Examining Division refused the then main request and the then auxiliary requests 1 and 2 for lack of clarity and for lack of inventive step over document D1.

III. With the statement of grounds of appeal, the appellant replaced its requests with an amended main request and amended auxiliary requests 1 and 2.

IV. In a communication under Article 15(1) RPBA following a summons to oral proceedings, the Board expressed its preliminary view that all requests lacked clarity and did not involve an inventive step in view of the common general knowledge of the skilled person.

V. With a letter dated 8 August 2016, the appellant filed an auxiliary request 3 and submitted further arguments.

VI. Oral proceedings were held on 8 September 2016. At the end of the oral proceedings, the chairman pronounced the Board's decision.

VII. The appellant requested that the decision under appeal be set aside and that a patent be granted on the basis of the claims of the main request or, in the alternative, on the basis of the claims of one of auxiliary requests 1, 2 and 3.

VIII. Claim 1 of the main request reads as follows:

"A method of providing data from an online transactional processing system (200) to a data sink (210), the method comprising the steps of:

a) receiving (300) a first request (226) from the data sink,

b) in response to the first request, sending (302) of a list (228) of entities of the online transactional processing system (200) to the data sink,

c) receiving (304) a second request (232) from the data sink, the second request being indicative of at least one of the entities (E_i),

d) in response to the second request, sending (306) of a first descriptor (234; FN_1, FN_2, FN_3,...) being descriptive of an extractor structure (412) of the at least one entity of the online transactional processing system (200) to the data sink,

e) receiving (308) of a data request (236) from the data sink, the data request being indicative of the at least one entity and comprising a second descriptor being descriptive of a sub-structure (414; FN_2, FN_3) of the extractor structure of the at least one entity,

f) filling (310) the extractor structure with data from the online transactional processing system (200),

g) reducing (312) the extractor structure to the sub-structure,

h) sending (314) of the data (238) of the sub-structure to the data sink."

IX. Claim 1 of auxiliary request 1 differs from claim 1 of the main request in that the following text has been added at the end of step e):

"the data request being a third remote function call for calling a third function (218) which serves to invoke an extractor for filling the extractor structure with data,"

and in that features g) and h) have been renamed h) and i) and a new feature g) has been inserted reading as follows:

"g) providing the sub-structure in the form of a sequential list (416),"

X. Claim 1 of auxiliary request 2 differs from claim 1 of auxiliary request 1 in that the following text has been added at the end of the claim:

"the first descriptor comprising at least the field names (FN_1, FN_2, FN_3,...) of the at least one entity the second descriptor comprising a sub-set (FN_2, FN_3) of the field names."

XI. Claim 1 of auxiliary request 3 reads as follows:

"A method of providing data from an online transactional processing system (200) to a data sink (210), the method comprising the steps of:

a) receiving (300) a first request (226) from the data sink,

b) in response to the first request, sending (302) meta data being indicative of a list (228) of entities of the online transactional processing system (200) to the data sink,

c) receiving (304) a second request (232) from the data sink, the second request being indicative of at least one of the entities (E_i),

d) in response to the second request, sending (306) of a first descriptor (234; FN1, FN2, FN3,...) being descriptive of an extractor structure (412) of the at least one entity of the online transactional processing system (200) to the data sink, the extractor structure being tabular, the first descriptor comprising at least the field names of the extractor structure;

e) receiving (308) of a data request (236) from the data sink, the data request being indicative of the at least one entity and comprising a second descriptor being descriptive of a sub-structure (414; FN2, FN3) of the extractor structure of the at least one entity, the second descriptor comprising field names (FN_j) specifying the sub-structure of the extractor structure, said field names in the second descriptor being a sub-set of the ones of the field names of the extractor structure which are of interest to the data sink, the data request being a remote function call for calling a function (218) which serves to invoke the extractor for filling the extractor structure with data, the extractor being a program used to fill the extractor structure with data from the online transactional processing system (200);

f) filling (310) the extractor structure with the data extracted from the online transactional processing system (200),

g) reducing (312) the extractor structure to the sub-structure;

h) providing the data of the sub-structure in the form of a sequential list (416) by transforming the sub-structure filled with the data into the sequential list, the sequential list having a generic format which is independent from the format of sub-structure (414), the list having one column for all field names (FN) and one column for all field values (FW_{jk});

i) sending (314) of the data (238) of the sub-structure in the form of the sequential list to the data sink,

the first descriptor comprising at least the field names (FN_1, FN_2, FN_3,...) of the at least one entity the second descriptor comprising a sub-set (FN_2, FN_3) of the field names."

XII. The appellant's arguments relevant to the decision are discussed in detail below.

Reasons for the Decision

1. The appeal complies with the provisions referred to in Rule 101 EPC and is therefore admissible.

2. *Interpretation of certain terms*

2.1 *The term "data sink"*

Claim 1 of the main request concerns a method of extracting data from an online transactional processing (OLTP) system to a data sink. Although the background section of the application discusses the providing of extracted data to a data warehouse, the description on page 4, lines 8 to 12, of the published application makes clear that the term "data sink" is intended to encompass any kind of application which requires data from a data source for any purpose.

2.2 *The term "list of entities"*

2.2.1 According to steps a) and b) of claim 1 of the main request, in response to a first request from the data sink, the OLTP system transmits to the data sink "a list of entities of the online transactional processing system".

In its decision the Examining Division considered that, in view of the application as a whole, the term "entity" referred to a database table held in a relational OLTP database.

2.2.2 The appellant explained that the term "entity" was to be understood as referring to a type of objects (such as employees or customers) about which data was stored in the form of attributes. A particular entity could correspond to exactly one database table, in which case the table rows corresponded to object instances (particular employees or particular customers) and table columns corresponded to attributes. But the data of an entity could also be stored in a plurality of database tables or other kinds of data structures.

2.2.3 The Board observes that the appellant's position on this point does not contradict that of the Examining Division: the term "entity" at least encompasses the interpretation that the Examining Division has given to it. But the appellant's point is that the scope of the term "entity" is not limited to database tables, and the Board sees no reason to disagree. An entity within the meaning of claim 1 may correspond to a particular physical database table held in the relational OLTP database, but its data may also be stored in other forms or, for example, be held in two or more database tables.

2.2.4 As to the term "list of entities", it is clear from claim 1 as a whole and also from the description on page 7, lines 25 to 29, that the OLTP system returns a list of the "available" data entities, i.e. data indicative of the entities about which data may be retrieved from the OLTP. Such a list may take the form of a list of entity names. If an entity corresponds to a database table, the entity name may be the name of the table.

2.3 *The term "descriptor"*

2.3.1 According to steps c) and d) of claim 1 of the main request, in response to a second request from the data sink indicative of at least one of the entities, the OLTP system transmits to the data sink a "first descriptor being descriptive of an extractor structure of the at least one entity". In step e), the OLTP system receives from the data sink a data request again indicative of the least one entity and further comprising a "second descriptor being descriptive of a sub-structure of the extractor structure of the at least one entity".

2.3.2 In claim 1 of auxiliary request 2, the "first descriptor" has been further specified as comprising "at least the field names of the at least one entity" and the "second descriptor" as comprising "a sub-set of the field names". Thus, the first descriptor may be interpreted as describing the at least one indicated entity (or an "extractor structure" for it) in terms of its field or attribute names and the second descriptor then lists the subset of field or attribute names which the data sink requests from the OLTP system.

2.4 *The terms "extractor structure" and "extractor program"*

2.4.1 According to steps f), g) and h) of claim 1 of the main request, the OLTP system fills the "extractor structure" with data stored at the OLTP system, "reduces" the extractor structure "to the sub-structure" and returns the data of the sub-structure to the data sink.

2.4.2 According to page 3, lines 5 to 11, of the background section of the application, various "data extractors" are known in the prior art. Such a data extractor is a program used to fill a given "extractor structure" with data from a database. In the prior-art OLTP system described on page 2, line 27, to page 3, line 15, with reference to Figure 1, each extractor program provided by the system has an extractor structure which is filled with data from a data source upon invocation of the extractor. The data warehouse that invokes one of the extractors by sending a corresponding request to the OLTP system generates an identical empty extractor structure, which then serves to accommodate the data received from the OLTP system in response to the request.

In one embodiment described in the detailed description of the application on page 10, lines 1 to 11, extractor structures are present only on the side of the OLTP system. The data requested from the OLTP system and contained in the "extractor sub-structure" is transformed into a sequential table 416, which has a "generic format which is independent from the format of sub-structure 414". Claim 1 of the main request at least encompasses this embodiment.

2.4.3 At the oral proceedings, the Board suggested and the appellant agreed that, in the context of relational databases, a stored query defining a database view is one example of an "extractor program". Such a stored query defines a virtual table, i.e. a table not corresponding to any physical database table, by specifying how to obtain that virtual table as the result set of a query on one or more database tables. Such a "view" on the database may itself be queried, essentially as if it were a physical database table.

2.4.4 With this interpretation of the term "extractor program", the term "extractor structure" encompasses a temporary data structure in the memory of the OLTP system that, as specified by feature f), is filled, by executing the database query corresponding to the extractor program, with the data of the virtual table. According to feature g), this temporary stored tabular data structure is then "reduced" to the columns which were requested by the data sink. The data in the resulting reduced table is subsequently read out and transmitted to the data sink in accordance with feature h).

3. *Main request - inventive step*

3.1 At the priority date of the present application, OLTP systems comprising data sources in the form of online relational databases were well known in the art (see e.g. page 2, lines 27 to 29, of the background section of the published application). Relational databases store data in database tables consisting of rows and columns, the rows corresponding to data records and the columns to data fields (or "attributes"). Tables and columns are identifiable by name; for example, the table "Employees" may comprise columns "employee_id", "name", "salary", "function".

3.2 It was also well known to query a relational database for the data in a table corresponding to a specified subset of the columns of that table. For example, if a relational database contained the aforementioned table "Employees", it was commonplace to retrieve from this table the data in columns "employee_id" and "name" by means of the following query expressed in the SQL query language:

```
SELECT employee_id, name FROM Employees
```

Instead of being stored as a physical database table, the "Employees" entity could also correspond to a database view, i.e. a virtual table defined by a stored query that extracts the data of the "Employees" entity from one or more physical database tables. Such an

"Employees" view could then be queried in the same way as a physical database table.

3.3 The Board considers such well-known OLTP database queries as applied to database views to constitute a suitable starting point for assessing inventive step for the subject-matter of claim 1. Such a query takes the name of a database view and a subset of the column names of the corresponding virtual table, and it returns for each row of the virtual table the values contained in the specified columns. The appellant did not contest the existence of this common general knowledge.

The receipt at the OLTP system of a database query specifying a database view and a subset of the column names of the associated virtual table corresponds to step e) of claim 1, the "entity" being the database view.

The subject-matter of claim 1 therefore differs from this prior art in preliminary steps a) to d), which set out how the data sink obtains the information necessary to formulate the query, and in steps f) to h), which describe how the query is executed.

3.4 As to steps a) to d), the Board observes that in order to specify a query of the type discussed above, it is necessary to know the name of the database view and the names of its columns. In a situation where this information is not known to the data sink, it is obvious to query the OLTP system in accordance with steps a) to d). These steps mirror how a human user would retrieve the necessary knowledge from the database administrator, namely by first asking for the names of the available entities and then asking for the column names of the entities he is interested in.

The appellant argued that prior-art database clients necessarily included the knowledge about the names and structure of database tables required for formulating queries. However, document D1 confirms on pages 6 and 7 that relational database systems known at the priority date could in fact be queried for the required metadata ("to query the source database's data dictionary (catalog) and extract metadata that describes the tables, views, etc. of interest to the user"). In other words, it was known to dynamically discover the available database tables and views and their structure by querying the database system.

3.5 At the oral proceedings, the appellant focused mainly on steps f) to h). In particular, step f) required the extractor structure to be filled with the data of the entity, and step g) then reduced the structure to the data actually being requested. The claim hence required the extractor structure to be filled with more data than was needed. A standard SQL query, on the other hand, directly retrieved exactly the amount of data that the data sink requested.

The appellant argued that using valuable CPU power for extracting more data from a data source than was actually needed by a client was counter-intuitive and contradicted many established principles of IT architecture. The technical benefit of that approach was that data extraction for a particular entity could be implemented generically for different data sinks or for different needs by the same data sink. The claimed invention therefore involved an inventive step.

3.6 The Board notes, however, that steps f) and g) correspond precisely to how a query of an entity in the form of a database view conceptually works.

As explained in point 2.4.3 above, a database view is defined by a stored query which, when executed, queries one or more database tables and thereby defines a virtual table. The stored query is the "extractor program" and if that query were actually executed, it would construct an intermediary table in the memory of the database server comprising the data corresponding to all the columns of the entity, before returning that data to the data sink. Querying the entity, i.e. the database view, for the data of a subset of its columns corresponds, at least conceptually, to a query on that intermediary table and, again conceptually, defines a reduction of that intermediary table to the data in the specified subset of columns.

3.7 The prior art on file does not disclose a database system implementing a query on a database view in a way that literally corresponds to this conceptual description of how such a query works. And the Board has in fact little doubt that, at least in advanced database systems, such a query will generally be performed in a more efficient way.

Nevertheless, the alleged benefit of the approach set out in claim steps f) to h), namely the possibility of generically implementing a data extractor that may be used for different sinks and for different needs by the same data sink, does not depend on whether performing a query on a database view is implemented "naively" by first constructing a full intermediary table in memory containing all the data of the database view and then "reducing" that table or whether it is implemented in a more efficient way that avoids extracting more data from the physical database tables than needed. In either case, it is the stored query that implements the data extractor, and this stored query itself need not be adapted if another data sink or the same data sink issues a request for the data corresponding to a different subset of the columns of the database view.

3.8 Hence, starting from the closest prior art identified in point 3.3 above, the Board considers that the technical problem solved by steps f) to h) is that of implementing the execution of a database view query. The skilled person confronted with this problem would understand that, conceptually, such a query is to do what is described in point 3.6 above. The straightforward implementation of this conceptual approach corresponds to steps f) to h). Although the skilled person would be aware that this implementation will generally be inefficient, the acceptance of such a foreseeable disadvantage not offset by any unexpected technical advantage does not involve an inventive step.

3.9 The appellant's remaining arguments were directed against equating the term "list of entities" as used in claim 1 with a list of names of physical database tables rather than a list of names of a more abstract type of "entities". The invention provided a layer of abstraction which resulted in a decoupling of the data sink from the structure of the physical database tables storing the entity data. The data sink needed to be aware only of the field names of the extractor structure and required no adaptation if the physical table structure of the data source changed.

The Board has equated the "entities" of claim 1 not with database tables, but with well-known database views. Database views provide a layer of abstraction in that when the organisation of the physical database tables storing the data of a database view changes, a data sink can continue to query the view in the same way as before; only the "extractor program" needs to be changed, i.e. the stored query defining how the data of the view is to be extracted from the physical database tables.

The appellant's arguments therefore cannot refute the Board's inventive-step reasoning.

3.10 Thus the subject-matter of claim 1 lacks inventive step (Articles 52(1) and 56 EPC).

4. *Auxiliary request 1 - inventive step*

4.1 Claim 1 of auxiliary request 1 adds to claim 1 of the main request that the data request of step e) is "a third remote function call for calling a third function which serves to invoke an extractor for filling the extractor structure with data". This feature corresponds to the additional feature of dependent claim 6 of the main request, dependent claims 2 and 3 introducing "first" and "second" remote function calls.

In addition, claim 1 renames steps g) and h) as h) and i) and adds a new step "g) providing the sub-structure in the form of a sequential list".

At the oral proceedings, the appellant agreed that new step g) should have been inserted after what is now step h).

4.2 Step g) indicates that the data requested by the client is transmitted to it in the form of a sequential list.

The appellant argued that known database interfaces returned a result set with three columns if three columns were specified in the query and with five columns if five columns were specified. The "sequential list" feature allowed "implementing a very generic interface for returning result data to the data sink".

The Board agrees with the appellant that the term "sequential list" is very generic. Since the example of a known result set given by the appellant, namely a result set comprising rows of data, each row consisting of a number of column values, may be viewed as a sequential list of data rows, new step g) does not add anything inventive.

4.3 Amended step e) now clarifies that the extractor structure is filled with data by an "extractor". This is in line with the Board's interpretation of claim 1 of the main request (see points 2.4.3 and 2.4.4 above).

4.4 Step e) further states that the extractor is invoked by the data sink by means of a "remote function call". The appellant acknowledged that remote function calls were a technique for making requests over a network connection that had been well-known in the art at the priority date of the application. The appellant essentially argued that it was not known to employ remote function calls to invoke the specific functionality of the data extraction logic of the invention. In addition, the use of remote function calls improved data security.

The Board considers, however, that the applicability of the known remote function call technique does not depend on the specific functionality that is being invoked. It further cannot agree that the use of remote function calls per se improves data security, which is an effect that is in any case not mentioned in the application as filed. Since the feature added to step e) merely proposes using a known technique for its normal purpose, the feature does not render the claimed subject-matter inventive.

4.5 Hence, the subject-matter of claim 1 of auxiliary request 1 lacks inventive step (Articles 52(1) and 56 EPC).

5. *Auxiliary request 2 - inventive step*

5.1 Compared to claim 1 of auxiliary request 1, claim 1 of auxiliary request 2 clarifies that the "first descriptor" comprises the field names of the at least one entity in which the data sink is interested and that the "second descriptor" comprises a subset of those field names.

5.2 Since these clarifying amendments are in line with the Board's interpretation of claim 1 of the main request and of auxiliary request 1 (see point 2.3.2 above), the subject-matter of claim 1 of auxiliary request 2 likewise lacks inventive step (Articles 52(1) and 56 EPC).

6. *Auxiliary request 3 - inventive step*

6.1 Apart from changes (and some repetition) in wording for the purpose of further clarification, claim 1 of auxiliary request 3 adds to claim 1 of auxiliary request 2 the following features:

- the extractor structure is tabular; and

- the sequential list has a generic format, independent from the format of the sub-structure, the list having one column for all field names and one column for all field values.

6.2 That the extractor structure is tabular is again in accordance with the Board's interpretation of claim 1 of the higher-ranking requests (see point 2.4.4 above).

6.3 The feature specifying that the sequential list has one column for all field names and one column for all field values is known from document D3. This document discusses in section 1.1 the conventional "horizontal" scheme of representing data objects as rows of a table, each object attribute corresponding to a column, and discloses in section 1.2 an alternative "vertical" representation in the form of a table comprising an "object identifier" column, an "attribute name" column and an "attribute value" column. The Board notes that Figure 4 of the present application, which illustrates the claimed "sequential list", similarly includes an "object identifier" column.

Since the application is silent on specific advantages that this known sequential-list representation may have in the context of the invention and the appellant did not further argue this point, the Board concludes that the subject-matter of claim 1 of auxiliary request 3 too does not involve an inventive step (Articles 52(1) and 56 EPC).

7. *Conclusion*

Since none of the appellant's requests is allowable, the appeal is to be dismissed.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

Beschwerdekammern des Europäischen Patentamts
Boards of Appeal of the European Patent Office
Chambres de recours de l'Office européen des brevets

I. Aperribay R. Moufang

Decision electronically authenticated