



(22) Date de dépôt/Filing Date: 2004/05/27

(41) Mise à la disp. pub./Open to Public Insp.: 2004/11/27

(45) Date de délivrance/Issue Date: 2010/06/29

(30) Priorité/Priority: 2003/05/27 (CA2,429,910)

(51) Cl.Int./Int.Cl. *G06F 17/30* (2006.01)

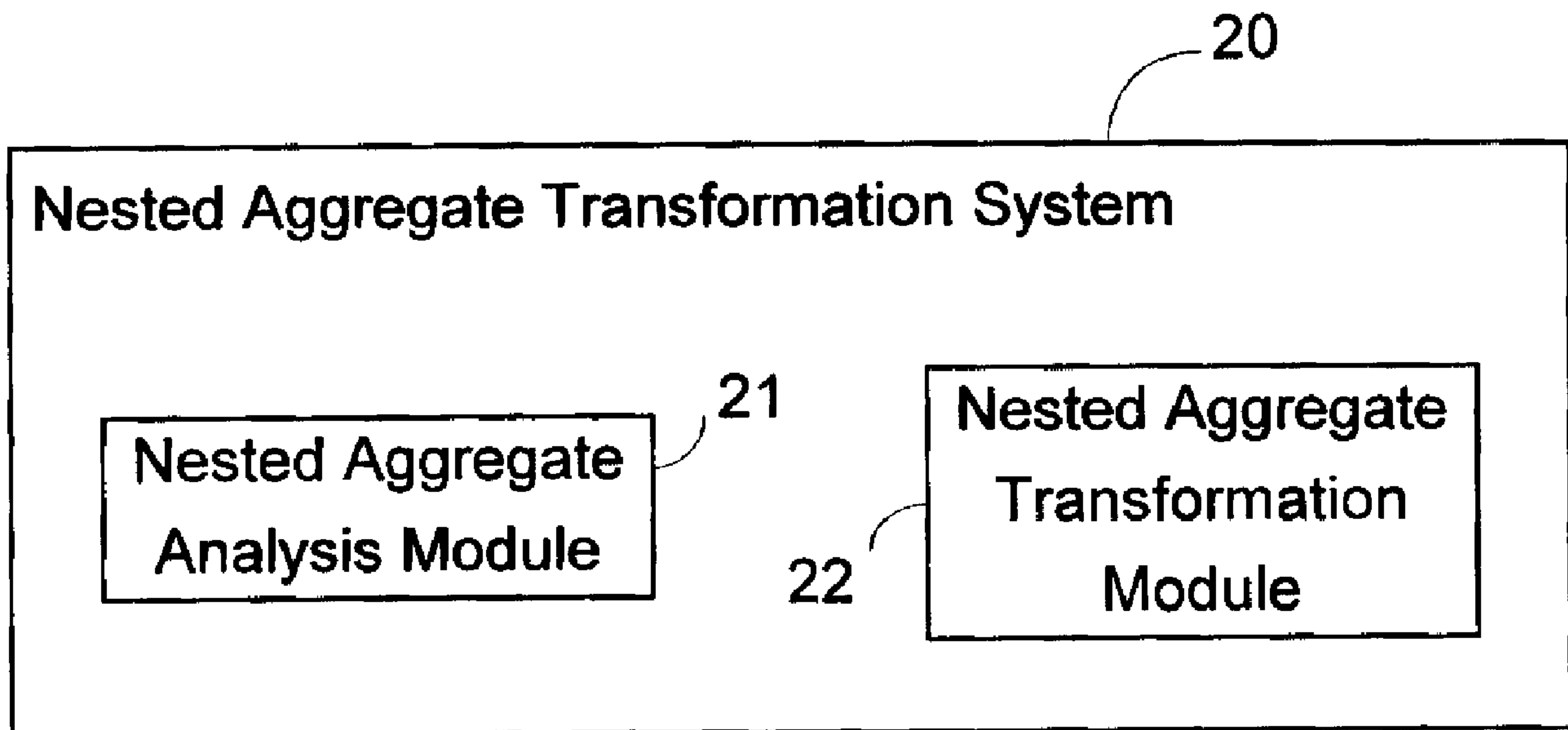
(72) Inventeur/Inventor:
STYLES, MICHAEL E., CA

(73) Propriétaire/Owner:
INTERNATIONAL BUSINESS MACHINES
CORPORATION, US

(74) Agent: WANG, PETER

(54) Titre : SYSTÈME ET METHODE DE TRANSFORMATION DES INTERROGATIONS

(54) Title: SYSTEM AND METHOD OF QUERY TRANSFORMATION



(57) Abrégé/Abstract:

A query transformation system for transforming nested aggregates in a query is provided. The query transformation system comprises a nested aggregate analysis module for analysing a query that is not supported by a target database system, and a nested aggregate transformation module for transforming the query into a semantically equivalent query that is supported by the target database system.

Abstract

A query transformation system for transforming nested aggregates in a query is provided. The query transformation system comprises a nested aggregate analysis
5 module for analysing a query that is not supported by a target database system, and a nested aggregate transformation module for transforming the query into a semantically equivalent query that is supported by the target database system.

System and Method of Query Transformation

FIELD OF THE INVENTION

The invention relates generally to data access middleware, and in particular to a
5 system and method of query transformation.

BACKGROUND OF THE INVENTION

A typical data access environment has a multi-tier architecture. For description
purposes, it can be separated into three distinct tiers:

- 10 • Web server
- Applications
- Data

The tiers are based on business function, and are typically separated by firewalls. Client
software, such as a browser or a report-authoring tool, sits above the tiers.

15 The web server contains a firewall and one or more gateways. All web
communication is performed through a gateway. A gateway is responsible for passing on
requests to the application server, in tier 2, for execution.

 The applications tier contains one or more application servers. The application
server runs requests, such as reports and queries that are forwarded by a gateway running
20 on the web server. Typically, one of the components of the applications tier is a query
engine, which is data access middleware that provides universal data access to a variety
of heterogeneous database systems. The query engine formulates queries (typically SQL)
and passes them on to the data tier, through a native database API (such as ODBC) for
execution.

25 The data tier contains database management systems (DBMS), which manage raw
data stored in a database. Examples of such systems include Oracle, DB2, and Microsoft
SQL Server.

 Although a multi-tier architecture can be configured in several different ways, a
typical configuration places each tier on a separate computer (server). A database server
30 is typically a “high end” server, and thus can process queries at a relatively fast speed.
An application server cannot generally process queries as quickly as a database server.

 In order to solve many business questions, a query engine may generate SQL
queries that utilize the SQL/OLAP technology introduced in the SQL:1999 standard.

However, many database systems do not support this technology. Thus, the SQL queries would have to be performed by the query engine on the application server that is generally slower than the database server. It is desirable to have as much processing performed on the database server.

5 There is a need to prevent or reduce the amount of local (application server) processing required to process a query. In the past, the application would be responsible for generating SQL that contained no nested OLAP functions. Quite often, generating this type of SQL is more difficult since it is more complex.

10 One way of overcoming this problem is for the query engine to generate a simple base query for collecting the data, and then process the remainder of the query (including OLAP functions) to produce the desired result. Unfortunately, this solution requires extensive processing time on the application server. It is desirable to have a way of transforming the query into a form that minimizes the processing time on the application server. Ideally, the transformed query would be processed by the DBMS in its entirety.

15

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a method of nested aggregate transformation in a database system that supports SQL/OLAP, but does not support nested OLAP functions.

20 In accordance with an embodiment of the present invention, there is provided a query transformation system for transforming nested aggregates in a query. The query transformation system comprises a nested aggregate analysis module for analysing a query that is not supported by a target database system, and a nested aggregate transformation module for transforming the query into a semantically equivalent query
25 that is supported by the target database system.

In accordance with another embodiment of the present invention, there is provided a method of nested aggregate transformation. The method comprises the steps of analysing a query that is not supported by a target database system, and transforming the query into a semantically equivalent query that is supported by the target database system.

30 In accordance with another embodiment of the present invention, there is provided a method of nested aggregate transformation. The method comprises the steps of generating a derived table to compute aggregates, extracting nested aggregates and moving nested aggregates into a select list of the derived table, and determining whether

the transformation is to be performed. The step of determining includes the steps of detecting nested aggregates, detecting framed aggregates and report aggregates, performing the transformation if nested aggregates are detected, and adding expressions to the inner select list of the derived table. The step of performing includes the steps of
 5 marking a nested aggregate for insertion into an inner select list, marking a report aggregate for insertion into the inner select list if it does not contain a nested aggregate or there are framed aggregates present, and detecting framed aggregates.

In accordance with another embodiment of the present invention, there is provided a computer data signal embodied in a carrier wave and representing sequences of
 10 instructions which, when executed by a processor, cause the processor to perform a method of nested aggregate transformation. The method comprises the steps of analysing a query that is not supported by a target database system, and transforming the query into a semantically equivalent query that is supported by the target database system.

In accordance with another embodiment of the present invention, there is provided
 15 a computer-readable medium having computer readable code embodied therein for use in the execution in a computer of a method of nested aggregate transformation. The method comprises the steps of analysing a query that is not supported by a target database system, and transforming the query into a semantically equivalent query that is supported by the target database system.

In accordance with another embodiment of the present invention, there is provided
 20 a computer program product for use in the execution in a computer of a query transformation system for transforming nested aggregates in a query. The computer program product comprises a nested aggregate analysis module for analysing a query that is not supported by a target database system, and a nested aggregate transformation
 25 module for transforming the query into a semantically equivalent query that is supported by the target database system.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 shows a typical data access environment.

Figure 2 shows a nested aggregate transformation system, in accordance with an
 30 embodiment of the present invention.

Figure 3 shows in a flowchart an example of a method of a nested aggregate transformation, in accordance with an embodiment of the nested aggregate transformation system.

Figure 4 shows in a flowchart another example of a method of a nested aggregate transformation, in accordance with an embodiment of the nested aggregate transformation system.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Figure 1 shows a typical data access environment 10 for processing data.

Typically, data is stored in a database 11. A database management system (DBMS) running on a database server 12 accesses the raw data stored in the database 11. A query engine 15, running on a report server (or application server) 13 is used to generate reports on the raw data and instruct the DBMS on the database server 12 to obtain information pertaining to the raw data in the database 11. The query engine 15 provides universal data access to a variety of heterogeneous database systems. An end user uses a client application 14, running on a client workstation, to facilitate application server 13 operations.

In order to solve many business questions, a query engine 15 generates SQL queries that utilize the SQL/online analytical programming (OLAP) technology introduced in the SQL:1999 standard. These SQL queries include SQL/OLAP functions (windowed aggregates). However, many database systems 12 do not support this technology. In order to prevent or reduce the amount of local (application server) processing required to process these types of queries, the query engine 15 attempts to generate semantically equivalent queries that can be processed on the database server 12 by the target database system. These semantically equivalent queries included standard aggregate functions and a GROUP BY operator.

In order to solve many business questions, SQL (Structured Query Language) queries are generated that utilize the SQL/OLAP (Online Analytical Programming) technology introduced in the SQL:1999 standard. Quite often, these queries contain OLAP functions whose arguments and/or window specifications contain other (nested) OLAP functions. The SQL:1999 standard (and most databases with a SQL/OLAP capability) does not permit this.

Figure 2 shows a nested aggregate transformation system 20 for generating a query with no nested OLAP functions that is semantically equivalent to a query having nested OLAP functions. The nested aggregate transformation system 20 comprises a nested aggregate analysis module 21 for analyzing SQL/OLAP queries containing nested OLAP functions that are not supported by a target database system, and a nested aggregate transformation module 22 for transforming these SQL/OLAP queries into semantically equivalent queries that are supported by the target database system. The nested aggregate analysis module 21 may include a nested aggregate detection unit for traversing a query and detecting a nested aggregate in the query, and a nested aggregate tagging unit for traversing a query and tagging the nested aggregate. The nested aggregate transformation module may include a derived table for storing a copy of the tagged nested aggregate, and a query editor for inserting the tagged nested aggregate into a query that is supported by a target database system.

The nested aggregate transformation system 20 may be implemented as a subsystem of the query engine 15 in the data access environment 10. This transformation 20 may generate queries that can be processed in their entirety on the database server 12, or queries that require processing on both the application server 13 and the database server 12.

Figure 3 shows in a flowchart an example of a method of nested aggregate transformation (30), in accordance with an embodiment of the nested aggregate transformation system 20. The method (30) begins with analyzing a query containing a nested aggregate functions that is not supported by a target database system (31). Next, the query is transformed into a semantically equivalent query that is supported by the target database system (32). The method (30) is done (33).

Advantageously, the nested aggregate transformation system 20 eases the task of generating SQL by eliminating the need to analyze all aggregation required by the report, and by eliminating the need to generate nested derived tables. Furthermore, the nested aggregate transformation system 20 allows the query engine 15 to only perform the transformation if necessary.

There are two types of OLAP functions: framed functions and report functions. Framed OLAP functions contain a window frame specification (ROWS or RANGE) and an ORDER BY clause. Through window frames, capabilities such as cumulative

(running) sums and moving averages can be supported. Report functions do not contain a window frame specification, and produce the same value for each row in a partition.

A nested aggregate/OLAP function is any aggregate/OLAP function appearing inside the specification of another aggregate/OLAP function. These aggregates/OLAP functions may appear in the operand, the PARTITION BY clause, the AT (compute break) clause, or the ORDER BY clause. Nested aggregates are computed prior to the parent aggregate being computed. Framed OLAP functions are given special attention when performing the nested aggregate transformation 20 due to the presence of the ORDER BY clause.

Figure 4 shows a flowchart of an example of a method of nested aggregate transformation (40), in accordance with an embodiment of the nested aggregate transformation system 20. The method (40) begins with searching for nested aggregates and framed aggregates in a select list (41). If nested aggregates are not found (42), then the method is done (46). If nested aggregates are found (42), then the select list is traversed again with tags added to appropriate aggregates (43). A derived table is created (44). Once again, the select list is traversed and simple column references and tagged aggregates are added to the select list of the derived table (45). The method (40) is done (46).

The nested aggregate transformation system 20 generates a derived table to compute these aggregates. Nested aggregates are extracted and moved into the select list of the derived table. This process may be repeated several times, depending on the level of nesting.

To determine whether the transformation must be performed, each expression in the select list is analyzed to detect the presence (if any) of nested aggregates (31). This analysis comprises the following:

- Detection of nested aggregates.
- Detection of framed aggregates (nRAggregates) and report aggregates (nXAggregates).

If nested aggregates are detected (32), the transformation is performed.

The transformation performs a second analysis of the select list to determine how the transformation should be performed. This analysis comprises:

- Marking a nested aggregate for insertion into the inner select list.

- Marking a report aggregate for insertion into the inner select list if it does not contain a nested aggregate or there are framed aggregates present ($nRAggregates > 0$).
- Detection of framed aggregates.

5 Once this is complete, the select list is traversed again, with the following types of expressions being added to the inner select list:

1. Simple column references.
2. Aggregates marked for insertion. For all other aggregates, the AT, PARTITION BY, and ORDER BY clauses are processed according to the rules outlined here.
3. Expressions that do not contain nested aggregates or framed aggregates. All other expressions are traversed, and processed according to the rules outlined here.

15 Once the nested aggregate transformation is performed, other query transformations can be performed as required.

Example 1

In this example, a nested OLAP function appears in the operand of another OLAP function, so the query must be transformed.

20 Original Query

```
SELECT  MAX( SUM( QTY ) OVER ( PARTITION BY SNO ) ) OVER ()
FROM    SUPPLY
```

Transformed Query

```
25 SELECT  MAX( C0 ) OVER ()
FROM ( SELECT  SUM( QTY ) OVER ( PARTITION BY SNO ) C0
        FROM    SUPPLY ) T1
```

Explanation

30 Against Oracle9i, the original query will result in an error since the OLAP function MAX contains a nested SUM function in its operand. To eliminate the nesting, a derived table T1 is created, and the SUM function is pushed into the select list.

Example 2

In this example, a nested OLAP function appears in the operand of an OLAP function. As well, the same nested OLAP function appears on its own in the select list. This query must be transformed.

5 **Original Query**

```
SELECT  SNO, PNO, SUM( QTY ) OVER ( PARTITION BY SNO ),
        MAX( SUM( QTY ) OVER ( PARTITION BY SNO ) ) OVER ()
FROM    SUPPLY
```

10 **Transformed Query**

```
SELECT  C0, C1, C2, MAX( C2 ) OVER ()
FROM (   SELECT  SNO C0, PNO C1,
                SUM( QTY ) OVER ( PARTITION BY SNO ) C2
        FROM    SUPPLY ) T1
```

15

Explanation

Against Oracle9i, the original query will result in an error since the OLAP function MAX contains a nested SUM function in its operand. To eliminate the nesting, a derived table T1 is created, and the SUM function is pushed into the select list. Note also that a small optimization is performed with respect to the first OLAP function (SUM) in the select list of the original query, since it is identical to the operand of the MAX function. The SUM function only needs to be computed once.

Example 3

25 In this example, a nested OLAP function appears in the operand of an OLAP function. As well, the nested OLAP function itself contains a nested OLAP function in its PARTITION BY clause. This query must be transformed.

Original Query

```
SELECT  SNO, PNO,
30      SUM( QTY ) OVER ( PARTITION BY
                        AVG( QTY ) OVER ( PARTITION BY JNO ) ),
        MAX( SUM( QTY ) OVER ( PARTITION BY
                        AVG( QTY ) OVER ( PARTITION BY JNO ) ) ) OVER ()
```


FROM SUPPLY

Transformed Query

Pass 1

5 SELECT C0, C1, C2, MAX(C2) OVER ()
FROM (SELECT C0, C1,
SUM(C2) OVER (PARTITION BY
AVG(QTY) OVER (PARTITION BY JNO)) C2
FROM SUPPLY) T0

10

Pass 2

SELECT C0, C1, C2, MAX(C2) OVER ()
FROM (SELECT C0, C1, SUM(C2) OVER (PARTITION BY C3) C2
FROM (SELECT SNO C0, PNO C1, QTY C2,
15 AVG(QTY) OVER (PARTITION BY JNO) C3
FROM SUPPLY) T0) T1

Explanation

Against Oracle9i, the original query will result in an error since the OLAP
20 functions SUM and MAX contain nested OLAP functions. This particular example
requires 2 passes. In the first pass, the SUM function is pushed into a derived table to
eliminate all nested aggregation in the top-level select list. The second pass, required
since the SUM function contains a nested AVG function in its PARTITION BY clause,
results in the AVG function being moved into another derived table.

25

Example 4

In this example, a nested OLAP function appears in the operand of an OLAP
function. As well, the select list contains a framed aggregate that computes a cumulative
SUM. This query must be transformed.

30

Original Query

SELECT SNO, PNO,
SUM(QTY) OVER(
ORDER BY SNO, PNO ROWS UNBOUNDED PRECEDING),


```

SUM( QTY ) OVER ( PARTITION BY SNO ),
MAX( SUM( QTY ) OVER ( PARTITION BY SNO ) OVER ()
FROM    SUPPLY

```

Transformed Query

5 Pass 1

```

SELECT  C0, C1,
        SUM( C2 ) OVER(
            ORDER BY C1, C2 ROWS UNBOUNDED PRECEDING ) C2,
        C3, C4
10 FROM ( SELECT  SNO C0, PNO C1, QTY C2
            SUM( QTY ) OVER ( PARTITION BY SNO ) C3,
            MAX( SUM( QTY ) OVER ( PARTITION BY SNO )
                OVER () C4
            FROM    SUPPLY ) T0

```

15

Pass 2

```

SELECT  C0, C1,
        SUM( C2 ) OVER( ORDER BY C1, C2
            ROWS UNBOUNDED PRECEDING ) C2,
20      C3, C4
FROM ( SELECT  C0, C1, C2, C3, MAX( C3 ) OVER() C4
      FROM ( SELECT  SNO C0, PNO C1, QTY C2,
                    SUM( QTY ) OVER ( PARTITION BY SNO ) C3
            FROM    SUPPLY ) T0 ) T1

```

25

Explanation

Against Oracle9i, the original query will result in an error since the OLAP function MAX contains a nested SUM function in its operand. This particular example requires 2 passes. In the first pass, the SUM and MAX functions are pushed into a

30 derived table since the main select list contains a framed aggregate. The framed aggregate is left in the main select list since it requires the data to be ordered a specific way to compute the value. If this aggregate was pushed into the inner select list, any sorting required to compute aggregates in the main select may disrupt the ordering of

values produced by the framed aggregate. The second pass, required since the MAX function contains a nested SUM function in its operand, results in the SUM function being moved into another derived table.

5 The systems and methods according to the present invention may be implemented by any hardware, software or a combination of hardware and software having the functions described above. The software code, either in its entirety or a part thereof, may be stored in a computer readable memory. Further, a computer data signal representing the software code that may be embedded in a carrier wave may be transmitted via a
10 communication network. Such a computer readable memory and a computer data signal are also within the scope of the present invention, as well as the hardware, software and the combination thereof.

 While particular embodiments of the present invention have been shown and described, changes and modifications may be made to such embodiments without
15 departing from the true scope of the invention.

WHAT IS CLAIMED IS:

1. A method of transforming a query containing a nested windowed aggregate, comprising:
 - analyzing the query to identify the nested windowed aggregate, the nested windowed aggregate appearing inside the specification of an outer aggregate function;
 - generating a derived table query to compute the nested windowed aggregate, comprising:
 - adding the nested windowed aggregate to a SELECT list of the derived table query; and
 - adding a table of a FROM list of the query to a FROM list of the derived table query;
 - replacing the nested windowed aggregate in the SELECT list of the query with a reference to the nested windowed aggregate in the SELECT list of the derived table query; and
 - replacing the table of the FROM list of the query to the derived table query.
2. The method as claimed in claim 1, wherein analyzing the query comprises:
 - detecting the nested windowed aggregate to determine that the query should be transformed; and
 - identifying the nested windowed aggregate for adding to the SELECT list of the derived table.
3. The method as claimed in claim 1, wherein generating the derived table further comprises:
 - adding simple column references of the SELECT list of the query to the SELECT list of the derived table query; and
 - adding expressions of the SELECT list of the query that do not contain nested windowed aggregates to the SELECT list of the derived table query.
4. The method as claimed in claim 3, further comprising:
 - replacing the simple column references of the SELECT list of the query with references to the simple column references of the derived table query; and

replacing expressions that do not contain nested windowed aggregates in the SELECT list of the query with references to the expressions that do not contain nested windowed aggregates in the SELECT list of the derived table query.

5. The method as claimed in claim 1, further comprising repeating the analyzing, generating and replacing based on a level of nesting of aggregates in the query.
6. The method as claimed in claim 1, wherein the FROM list of the query comprises a plurality of tables;

adding the table of the FROM list of the query further comprises adding the plurality of tables of the FROM list of the query to the FROM list of the derived table query; and

replacing the table of the FROM list of the query comprises replacing the plurality of tables of the FROM list of the query to the derived table query.

7. A system for transforming a query containing a nested windowed aggregate, comprising:

a nested windowed aggregate analysis module for analyzing the query to identify the nested windowed aggregate, the nested windowed aggregate appearing inside the specification of an outer aggregate function; and

a nested windowed aggregate transformation module comprising:

a derived table module for generating a derived table query to compute the nested windowed aggregate, the derived table query generated by adding the nested windowed aggregate to a SELECT list of the derived table query and adding a table of a FROM list of the query to a FROM list of the derived table query; and

a query replacement module for replacing the nested windowed aggregate in the SELECT list of the query with a reference to the nested windowed aggregate in the SELECT list of the derived table query and replacing the table of the FROM list of the query to the derived table query.

8. The system as claimed in claim 7, wherein the nested windowed aggregate analysis module detects the nested windowed aggregate to determine that the query should be

transformed, and identifies the nested windowed aggregate for adding to the SELECT list of the derived table.

9. The system as claimed in claim 7, wherein the derived table generating module further adds simple column references of the SELECT list of the query to the SELECT list of the derived table query and adds expressions of the SELECT list of the query that do not contain nested windowed aggregates to the SELECT list of the derived table query.
10. The system as claimed in claim 9, wherein the query replacement module further replaces the simple column references of the SELECT list of the query with references to the simple column references of the derived table query, and replaces expressions that do not contain nested windowed aggregates in the SELECT list of the query with references to the expressions that do not contain nested windowed aggregates in the SELECT list of the derived table query.
11. The system as claimed in claim 7, wherein the system repeats the query transformation based on a level of nesting of aggregates in the query.
12. The system as claimed in claim 7, wherein the FROM list of the query comprises a plurality of tables;
 - the derived table module further for adding the plurality of tables of the FROM list of the query to the FROM list of the derived table query; and
 - the query replacement module further for replacing the plurality of tables of the FROM list of the query to the derived table query.
13. A computer-readable medium having computer readable code embodied therein for use in the execution of a method of transforming a query containing a nested windowed aggregate, the method comprising:
 - analyzing the query to identify the nested windowed aggregate, the nested windowed aggregate appearing inside the specification of an outer aggregate function;
 - generating a derived table query to compute the nested windowed aggregate, comprising:
 - adding the nested windowed aggregate to a SELECT list of the derived table query; and

adding a table of a FROM list of the query to a FROM list of the derived table query;

replacing the nested windowed aggregate in the SELECT list of the query with a reference to the nested windowed aggregate in the SELECT list of the derived table query; and

replacing the table of the FROM list of the query to the derived table query.

14. A computer program product for use in the execution in a computer of a system for transforming a query containing a nested windowed aggregate, comprising:

a nested windowed aggregate analysis module for analyzing the query to identify the nested windowed aggregate, the nested windowed aggregate appearing inside the specification of an outer aggregate function; and

a nested windowed aggregate transformation module comprising:

a derived table module for generating a derived table query to compute the nested windowed aggregate, the derived table query generated by adding the nested windowed aggregate to a SELECT list of the derived table query and adding a table of a FROM list of the query to a FROM list of the derived table query; and

a query replacement module for replacing the nested windowed aggregate in the SELECT list of the query with a reference to the nested windowed aggregate in the SELECT list of the derived table query and replacing the table of the FROM list of the query to the derived table query.

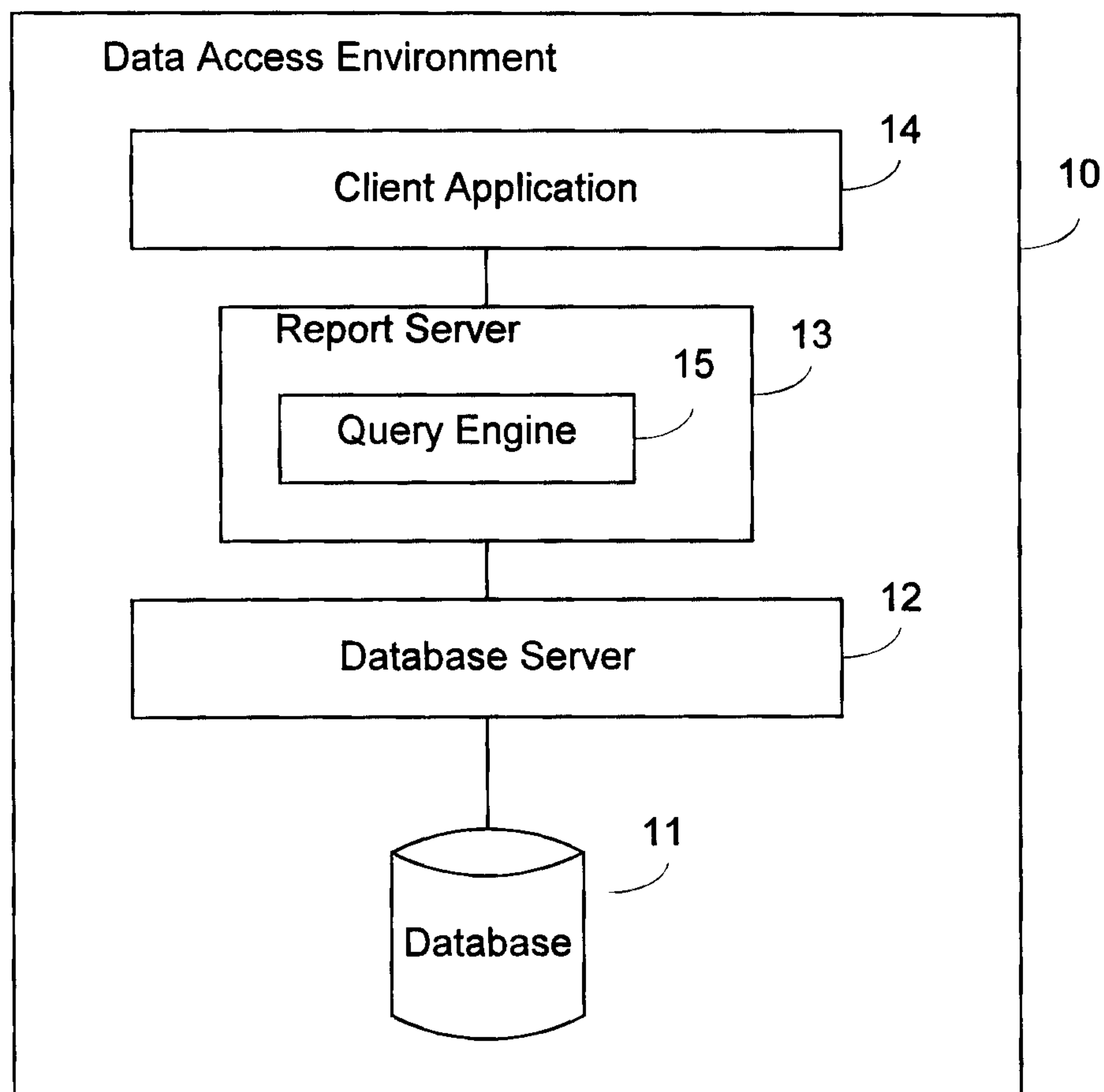


Figure 1

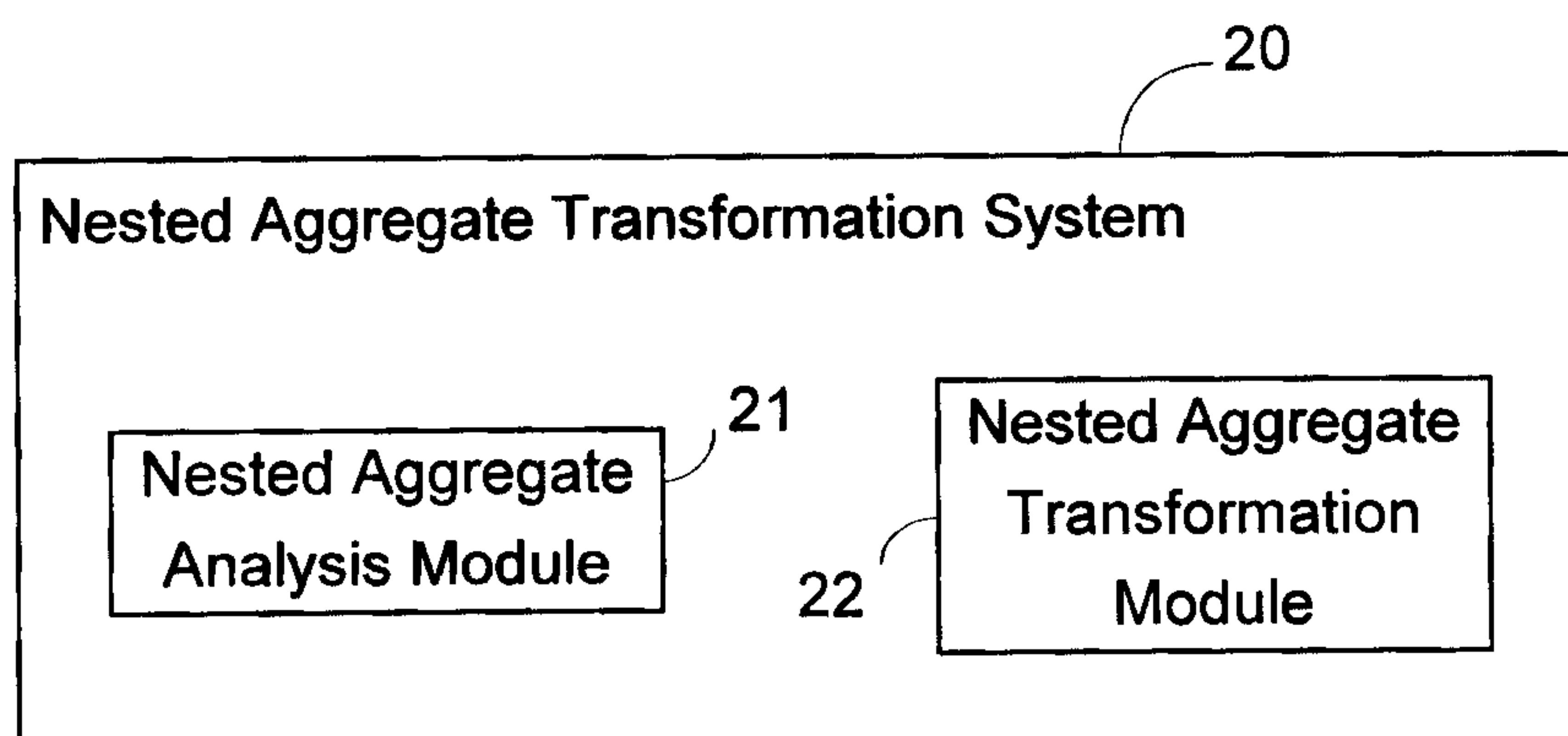


Figure 2

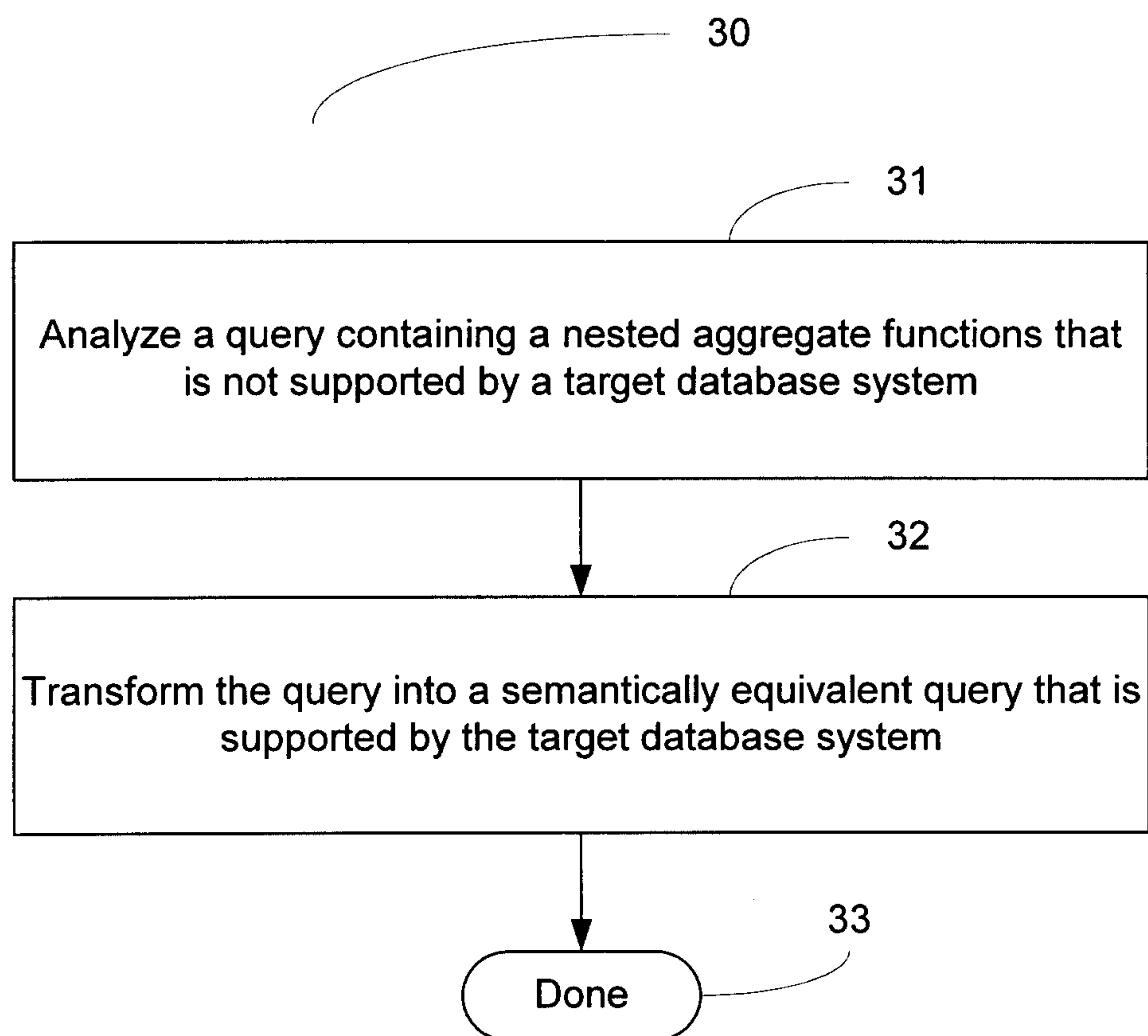
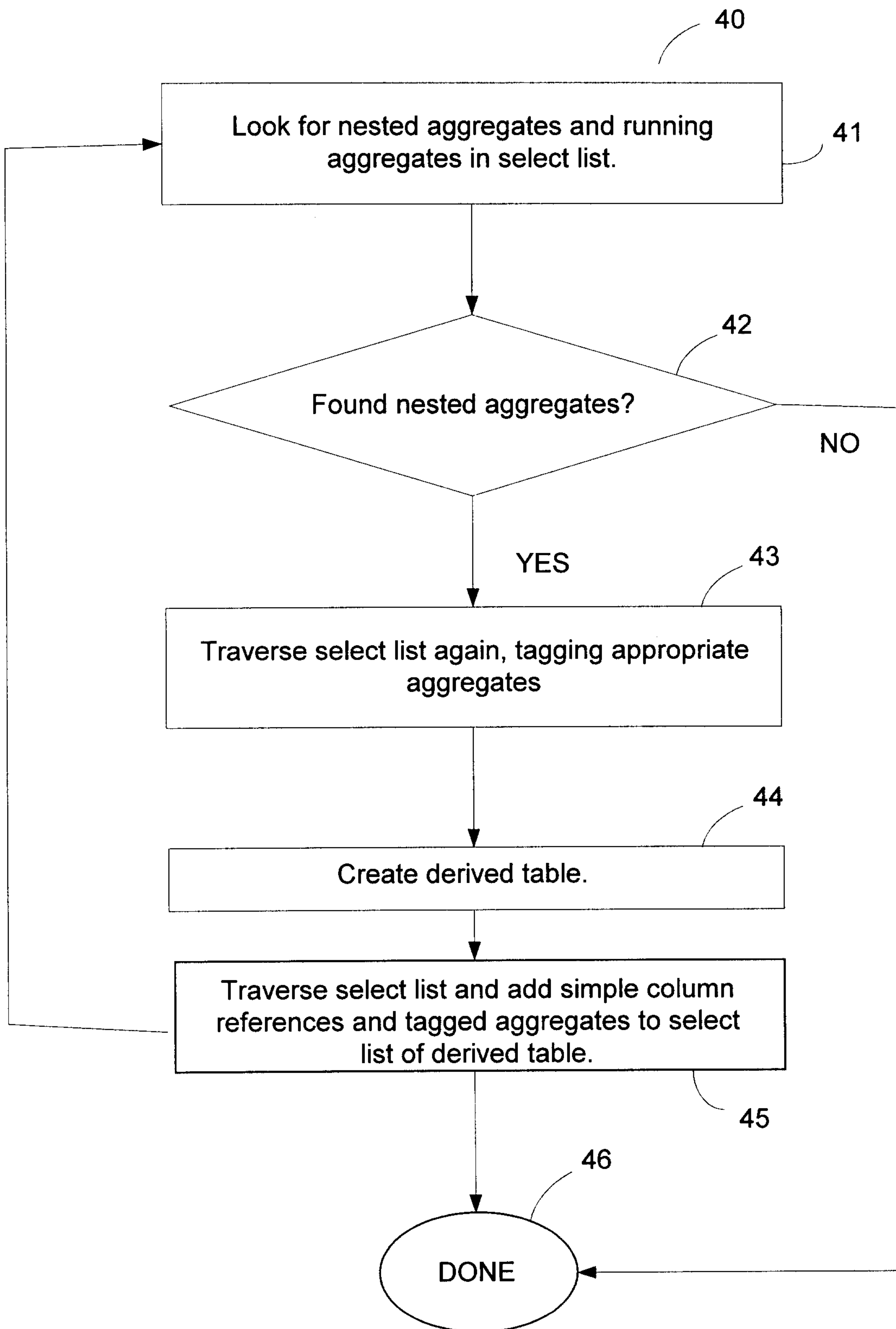


Figure 3

**Figure 4**

20

Nested Aggregate Transformation System

Nested Aggregate
Analysis Module

21

22

Nested Aggregate
Transformation
Module