Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2357937 A1 2003/03/28

(21) 2 357 937

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2001/09/28

(41) Mise à la disp. pub./Open to Public Insp.: 2003/03/28

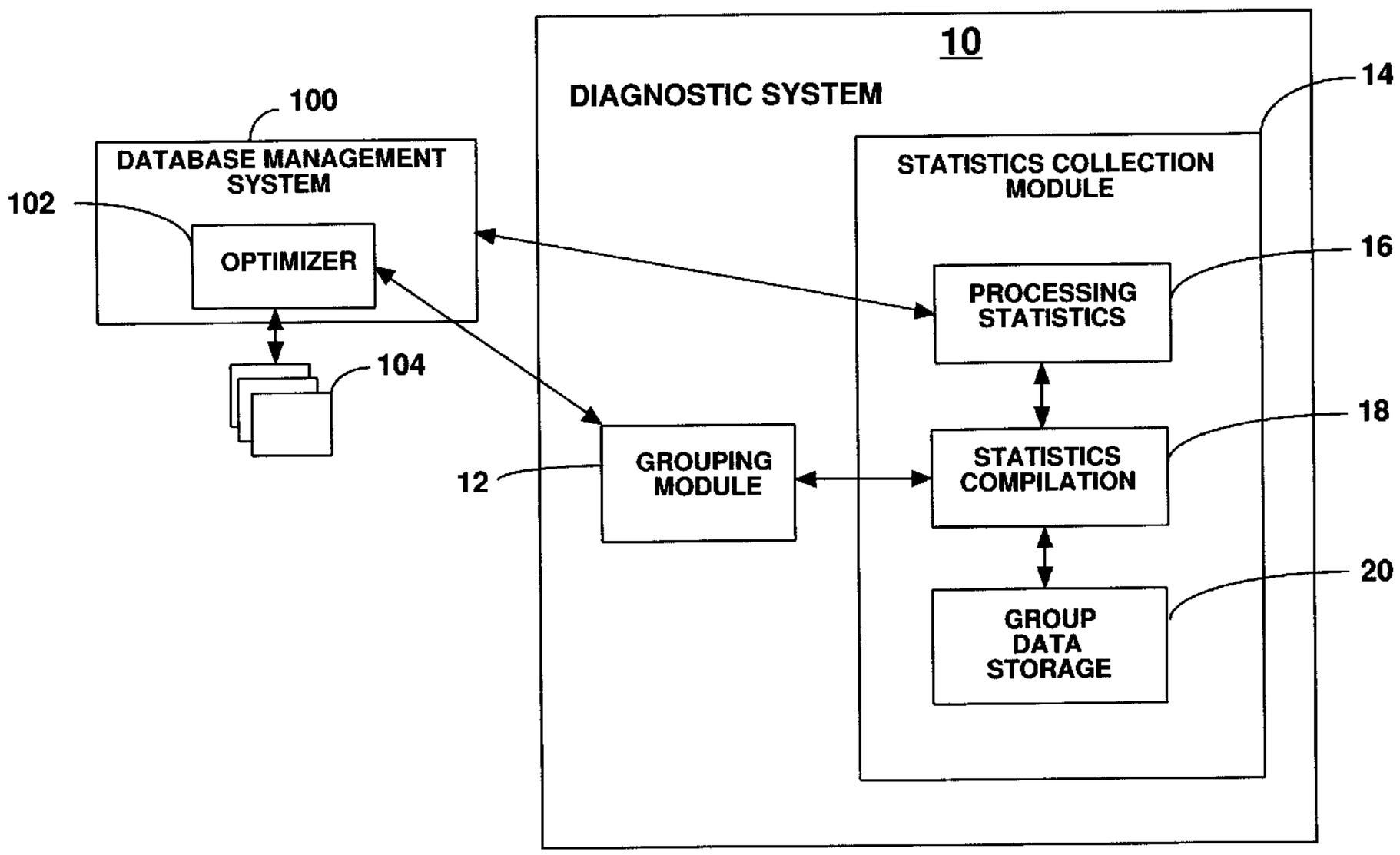
(51) Cl.Int.$^{7}$/Int.Cl.$^{7}$ G06F 17/30

(71) Demandeur/Applicant:
IBM CANADA LIMITED-IBM CANADA LIMITEE, CA

(72) Inventeurs/Inventors:
VALENTIN, GARY, CA;
HORMAN, RANDALL WILLIAM, CA;
LIGHTSTONE, SAM S., CA

(74) Agent: ROSEN, ARNOLD

(54) Titre : SYSTEME ET METHODE DE DIAGNOSTIC POUR BASES DE DONNEES
(54) Title: DATABASE DIAGNOSTIC SYSTEM AND METHOD




(57) Abrégé/Abstract:
A diagnostic tool for a database system. The diagnostic system includes a grouping module for assigning a grouping identifier to each query received by the database system; and a statistics collection module. The statistics collection module includes a query processing statistics module for obtaining processing statistics corresponding to each query; a group statistics compilation module for compiling processing statistics for each query by the query's grouping identifier; and group data storage for storing compiled processing statistics. A method is also disclosed for generating database diagnostic data.

## ABSTRACT OF THE DISCLOSURE

A diagnostic tool for a database system. The diagnostic system includes a grouping module for assigning a grouping identifier to each query received by the database system; and a statistics collection module. The statistics collection module includes a query processing statistics module for obtaining processing statistics corresponding to each query; a group statistics compilation module for compiling processing statistics for each query by the query's grouping identifier; and group data storage for storing compiled processing statistics. A method is also disclosed for generating database diagnostic data.

# DATABASE DIAGNOSTIC SYSTEM AND METHOD

## FIELD OF THE INVENTION

This invention relates to the field of database management systems generally, and in particular, to diagnostic tools for structured query language (SQL) database engines.

## BACKGROUND OF THE INVENTION

In the world of SQL database engines, seemingly small inefficiencies in processing an individual database query may rapidly become significant if the inefficiency affects thousands or tens of thousands of similar queries. As a result, diagnosing and resolving such inefficiencies can be of substantial importance. For greater clarity, while the term "query" is used throughout this document, it should be understood that this term is also intended to refer to any type of SQL statement, including statements that insert, delete or modify data. As will be understood by one skilled in the art, such statements are commonly referred to as "SQL statements" or DML (data manipulation language).

Although typically most database applications will receive at least some queries which share common attributes, for certain database applications many queries are of a standard or routine type. For example, in the banking industry, particularly with automated teller machines (ATMs) and credit card purchase processing, many routine SQL queries accessing basic account information will be substantially similar.

Accordingly, the applicants have recognized a need for a system and methodology for efficiently grouping similar database queries and compiling processing statistics for each such group for diagnostic purposes.

## SUMMARY OF THE INVENTION

The present invention is directed towards a diagnostic system for a database system.

The subject diagnostic system includes a grouping module for assigning a grouping identifier to each query received by the database system; and a statistics collection module. The statistics collection module in turn includes a query processing statistics module for obtaining processing

statistics corresponding to each query; a group statistics compilation module for compiling processing statistics for each query by the query's grouping identifier; and group data storage for storing compiled processing statistics.

The subject invention is also directed towards a program product stored on a computer readable medium. The program product includes a grouping module for assigning a grouping identifier to each query received by a database system; and a statistics collection module. The statistics collection module in turn includes a query processing statistics module for obtaining processing statistics corresponding to each query; a group statistics compilation module for compiling processing statistics for each query by the query's grouping identifier; and group data storage for storing compiled processing statistics.

The present invention is further directed towards a method of generating database diagnostic data comprising the following steps:

a. receiving a database query;

b. generating a query execution plan corresponding to the query;

c. generating a signature correlated to the query execution plan;

d. creating a group statistics table;

e. obtaining processing statistics to the query; and

f. storing the processing statistics in the group statistics table correlated to the signature.

## BRIEF DESCRIPTION OF THE DRAWINGS

The present invention will now be described, by way of example only, with reference to the following drawings, in which like reference numerals refer to like parts and in which:

Figure 1 is a schematic diagram of a database diagnostic system made in accordance with the present invention.

Figure 2A is a table of sample SQL queries.

Figure 2B is a chart illustrating a query execution plan corresponding to the first query of Figure 2A.

Figure 2C is a chart illustrating a binary coding corresponding to the query execution plan

of Figure 2B.

Figure 2D is a group data storage table of the diagnostic system of Figure 1.

Figure 3 is a flow chart showing a database diagnostic method employed using the diagnostic system of the present invention.

## DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Referring to Figure 1, illustrated therein is a preferred embodiment of the database diagnostic system of the subject invention. The system, shown generally as 10, comprises a grouping module 12 and a statistics collection module 14. The diagnostic system 10 is intended to interact with, and preferably is incorporated into, a database management system, shown generally as 100 which includes a query optimizer 102 and a database of data 104. In general, the diagnostic system 10 and database system 100 comprise software and data implemented on a hardware infrastructure.

As will be understood, the database management system 100 is configured to receive database queries (typically in the form of SQL) and return data in response to each query.

Figure 2A illustrates four different sample SQL queries which might be entered into an automobile registration database. The first query, Q1 is intended to determine how many red Mustangs are listed in the database. Similarly, the second query, Q2 is aimed at determining the number of silver Vipers in the database. The third query, Q3 is seeking the registered owner of a vehicle having the license plate number "ACTC403". The fourth query, Q4, determines the number of cars in the database.

In general, there may be numerous techniques for locating and retrieving the data requested by the query from the database 104. However, these different methods for obtaining the data are not uniformly efficient with respect to their use of resources (eg. processing time, memory, I/O (input/output) accesses). Accordingly, the database management system 100 is provided with a query optimizer 102 for determining an acceptably efficient method for retrieving the requested data, and generating a corresponding query execution plan (QEP). The QEP will typically be in the form of binary-encoded instructions (executed by the database system 100) for implementing a sequence of operations to resolve the SQL query.

There are two common types of optimizers 102: cost-based optimizers and rule-based

optimizers. Cost-based optimizers generate different possible QEPs for a query and estimate the processing cost for each possible QEP in terms of estimated resource usage (eg. CPU data processing time, data storage requirements). Generally the cost is reflected in a single number, and the QEP with the lowest cost is selected as the optimal QEP for answering the query.

In contrast, rule-based optimizers determine an optimal QEP for answering a query through reference to defined optimization rules. Depending on the nature of the query, different optimization rules will operate to generate the optimal QEP.

For greater clarity, it should be understood that while the term "optimal" is used herein in reference to the QEP generated by the optimizer 102, "optimal" is not intended to indicate that the generated QEP is the "best" in an objective sense.

Referring now to Figure 2B, illustrated therein is an example of a QEP, referred to as QEP1, in pseudo-code as generated by the optimizer 102 from the original query, Q1.

As will be understood, the function of the grouping module 12 is to identify similar SQL queries and group them together for analysis. Preferably, SQL queries are grouped together if they share certain predefined properties, such as: (a) they access data from the same data sources (eg. indexes, tables and materializations); (b) they are executed using substantially identical QEPs; and (c) they are executed in similar environments (eg. possess identical CPU parallelism; disk parallelism; disk constants - throughput, overhead; average buffering memory available; average number of connected applications; sort memory available; lock memory available; and CPU performance).

By appropriately grouping SQL queries together, each query in the group will preferably have the same data access patterns and will have the same patterns of resource consumption. As a result, processing inefficiencies affecting one query will likely similarly affect the entire group. Such inefficiencies may include insufficient bufferpool memory, incomplete indexing data, serialized I/O, and excessive sorting.

The grouping module 12 is coupled to receive data from the query optimizer 102 in order to generate a grouping identifier or signature to be assigned to the query.

The grouping module 12 may be configured to receive the optimal QEP from the optimizer 102. The optimizer 102 may be a cost-based or a rule-based optimizer. Once an optimal QEP has

been received, the grouping module 12 may then generate a grouping identifier or signature for the query by taking the first "word" of binary encoded instruction in the QEP and successively "folding" it into subsequent words, until a final signature word results.

By way of illustration, Figure 2C shows the binary coding B1 corresponding to the various operations O1 carried out by the database system 100, pursuant to the query execution plan, QEP1. As will be understood, generally the specific data sought by the query (eg. "Mustang" and "red") is not included in the binary coding of the QEP, unless patterns with respect to certain data have been noted and will serve as a useful grouping factor for diagnostic purposes.

As will be understood by one skilled in the art, although 6-bit words have been depicted for illustrative purposes in the binary coding B1, 32-bit words are commonly used in database management systems.

As noted above, the 6-bit words of the execution plan QEP1 are successively folded in together to form a signature or group identifier. The folding in may be accomplished by numerous different techniques including linear feedback shift register (LFSR) and CRC32 checksum, which involves applying the logical operator, XOR ("exclusive OR"), to the binary words. The result may be further refined (if necessary, depending on the application) by folding in identifiers for the tables and auxiliary objects (such as indexes and summary tables) which are accessed by the QEP during processing.

XORing the 6-bit words in the binary code B1, results in a binary signature S1 of 100111, which translates to 39 in decimal numerals. The following sample signatures correspond respectively to the queries Q1, Q2, Q3 and Q4 listed on Figure 2A

(S1): 100111 = 39

(S2): 100111 = 39

(S3): 010011 = 19

(S4): 000010 = 2

Because the signatures S1 and S2 are identical, the queries Q1 and Q2 may be considered sufficiently similar to be grouped together for diagnostic purposes. The queries S3 and S4 do not

match any of the other queries in the group, and accordingly the queries Q3 and Q4 will be grouped separately.

As will be understood, the degree of uniqueness of the signature for a query will be dependent upon the algorithm used by the grouping module 12 to generate and assign the signature. Accordingly, the uniqueness of the signature (and more specifically, the algorithm for generating the signature) may be adjusted as required for specific database systems 100.

If the optimizer 102 is a cost-based optimizer, instead of calculating the signature in the manner discussed above, the grouping module 12 may simply retrieve the resource cost calculated for the optimal QEP from the optimizer 102 and assign the cost as the signature/group identifier for the query Q1.

Referring back to Figure 1, the statistics collection module 14 preferably includes a query processing statistics module 16, a group statistics compilation module 18, and a group data storage 20. The query processing statistics module 16 is operatively coupled to the database system 100, to retrieve processing statistics related to actual resource usage by the database system 100 in resolving the query Q1.

The group statistics compilation module 18 compiles and stores the query processing statistics in the group data storage 20.

As shown in Figure 2D, the group data storage 20 stores query data correlated by group identifier 202. The stored data typically includes a sample SQL query 204 for the group, the number of queries 206 in the group, total CPU processing time 208 for all queries in the group, total sort time 210 for all queries in the group, total number of data read requests 212 and write requests 214. It should be noted that while only four common processing statistics 208, 210, 212, 214 have been selected for illustration on the data storage chart 20, for diagnostic purposes many different types of statistical data may be compiled relating to query processing. Such statistics often include: locks held, total sorts, index reads, index writes, rows inserted, rows deleted, rows updated and rows selected, but such listing is not intended to be comprehensive.

As will be understood, once a signature has been generated for a query, and the query has been resolved by the database system 100, the group statistics compilation module 18 retrieves the group identifier for the query from the grouping module 12, and also retrieves the processing

statistics 208 for the query from the database system 100. The compilation module 18 searches the data storage 20 to locate the group sharing the query's group identifier 202. Once the group is located, the number of queries data 206 is increased by one, and the other processing statistics for the query are added to the group statistics 208, 210, 212, 214. If no corresponding group is located, the data storage 20 creates a new group entry, storing the query as the group's sample query 204, and the query's processing statistics data 208, 210, 212, 214, as will be understood.

Referring briefly to the sample data on the data storage 20, the total CPU processing time 208 (and correspondingly the average CPU processing time for each query) for the group 220 having the group identifier "39" is markedly higher than for other groups. While not conclusive, such a disparity in resource usage might indicate a processing inefficiency.

Preferably, the compilation module 18 updates the data in the group data storage 20 in real time, ie. as each query is processed.

Figure 3 illustrates the steps of the method 300 to generate database diagnostic data carried out by the database system 100 and the diagnostic system 10 made in accordance with the subject invention.

The database 100 first receives a database query, for example, as a result of a user requesting a cash withdrawal from a bank account at an ATM. (Block 302) Upon receipt of the query, the optimizer 102 generates an optimal QEP corresponding to the query. (Block 304) The grouping module then generates a signature correlated to the QEP. The signature/group identifier may correspond to the cost assigned to the optimal QEP (if a cost-based optimizer is used), or may be calculated based on the operational steps set out in the QEP. (Block 306) In accordance with the QEP, the query is then processed and a response is returned. (Block 308) The query processing statistics module 16 then retrieves processing statistics data from the database system 100 for the query (Block 310), and the compilation module 18 compiles the processing data by group in the group data storage 20 (Block 312).

It should be understood that while the diagnostic system 10 could operate continuously with the operation of the database system 100, in general the diagnostic system 10 will only periodically be activated to compile diagnostic data.

Thus, while what is shown and described herein constitute preferred embodiments of the

subject invention, it should be understood that various changes can be made without departing from the subject invention, the scope of which is defined in the appended claims.

The embodiments of the invention in which an exclusive property or privilege is claimed are defined as follows:

1. A diagnostic system for a database system, the diagnostic system comprising:
   a. a grouping module for assigning a grouping identifier to each query received by the database system; and
   b. a statistics collection module including:
      i. a query processing statistics module for obtaining processing statistics corresponding to each query;
      ii. a group statistics compilation module for compiling processing statistics for each query by the query's grouping identifier; and
      iii. group data storage for storing compiled processing statistics.

2. The diagnostic system as claimed in claim 1, wherein the database system comprises a cost-based query optimizer for optimizing each query and assigning a projected processing cost to the query, and wherein the grouping module comprises a generator for generating a grouping identifier correlated to the cost for the query.

3. The diagnostic system as claimed in claim 1, wherein the database system comprises a query execution plan generation module for generating a query execution plan correlated to the query, and wherein the grouping module comprises a signature generation module configured to receive the query execution plan and to generate a corresponding grouping identifier.

4. The diagnostic system as claimed in claim 3, wherein the query execution plan generation module is a cost-based query optimizer.

5. The diagnostic system as claimed in claim 3, wherein the query execution plan generation module is a rule-based query optimizer.

6. The diagnostic system as claimed in claim 3, wherein the grouping module is configured to receive binary coding words corresponding to the query execution plan.

7. The diagnostic system as claimed in claim 6, wherein the grouping module is configured to fold the words together to generate the grouping identifier by the process of linear feedback shift register.

8. The diagnostic system as claimed in claim 6, wherein the grouping module is configured to fold the words together to generate the grouping identifier by the process of CRC32 checksum.

9. The diagnostic system as claimed in claim 6, wherein the grouping module is configured to fold the words together to generate the grouping identifier by applying the logical operator, XOR, to the words.

10. A program product stored on a computer readable medium, the program product comprising:

a. a grouping module for assigning a grouping identifier to each query received by a database; and

b. a statistics collection module including:

i. a query processing statistics module for obtaining processing statistics corresponding to each query;

ii. a group statistics compilation module for compiling processing statistics for each query by the query's grouping identifier; and

iii. group data storage for storing compiled processing statistics.

11. The program product as claimed in claim 10, wherein the database system comprises a cost-based query optimizer for optimizing each query and assigning a projected processing

cost to the query, and wherein the grouping module comprises a generator for generating a grouping identifier correlated to the cost for the query.

12. The program product as claimed in claim 10, wherein the database system comprises a query execution plan generation module for generating a query execution plan correlated to the query, and wherein the grouping module is configured to receive the query execution plan and to generate a corresponding grouping identifier.

13. The program product as claimed in claim 12, wherein the query execution plan generation module is a cost-based query optimizer.

14. The program product as claimed in claim 12, wherein the query execution plan generation module is a rule-based query optimizer.

15. The program product as claimed in claim 12, wherein the grouping module is configured to receive binary coding words corresponding to the query execution plan.

16. The program product as claimed in claim 15, wherein the grouping module is configured to fold the words together to generate the grouping identifier by the process of linear feedback shift register.

17. The program product as claimed in claim 15, wherein the grouping module is configured to fold the words together to generate the grouping identifier by the process of CRC32 checksum.

18. The program product as claimed in claim 15, wherein the grouping module is configured to fold the words together to generate the grouping identifier by applying the logical operator, XOR, to the words.

19. A method of generating database diagnostic data comprising the following steps:

a. receiving a database query;

b. generating a query execution plan corresponding to the query;

c. generating a signature correlated to the query execution plan;

d. processing the query;

e. obtaining processing statistics to the query; and

f. storing the processing statistics in the group statistics table correlated to the signature.

20. The method as claimed in claim 19, wherein step (f) comprises adding the processing statistics for the query to the processing statistics stored in the group statistics table for the group having a group identifier matching the signature.

21. The method as claimed in claim 19, wherein step (f) is performed substantially in real time.

22. The method as claimed in claim 19, wherein step (c) comprises generating binary coding words corresponding to the query execution plan and folding the words together to generate the group identifier.

23. The method as claimed in claim 22, wherein step (c) comprises folding the words together by the process of linear feedback shift register.

24. The method as claimed in claim 22, wherein step (c) comprises folding the words together by the process of CRC32 checksum.

25. The method as claimed in claim 22, wherein step (c) comprises folding the words together by applying the logical operator, XOR, to the words.

26. The program product as claimed in claim 10 wherein said computer readable medium

comprises a modulated carrier signal.

27. The program product as claimed in claim 10 wherein said computer readable medium comprises a storage medium.

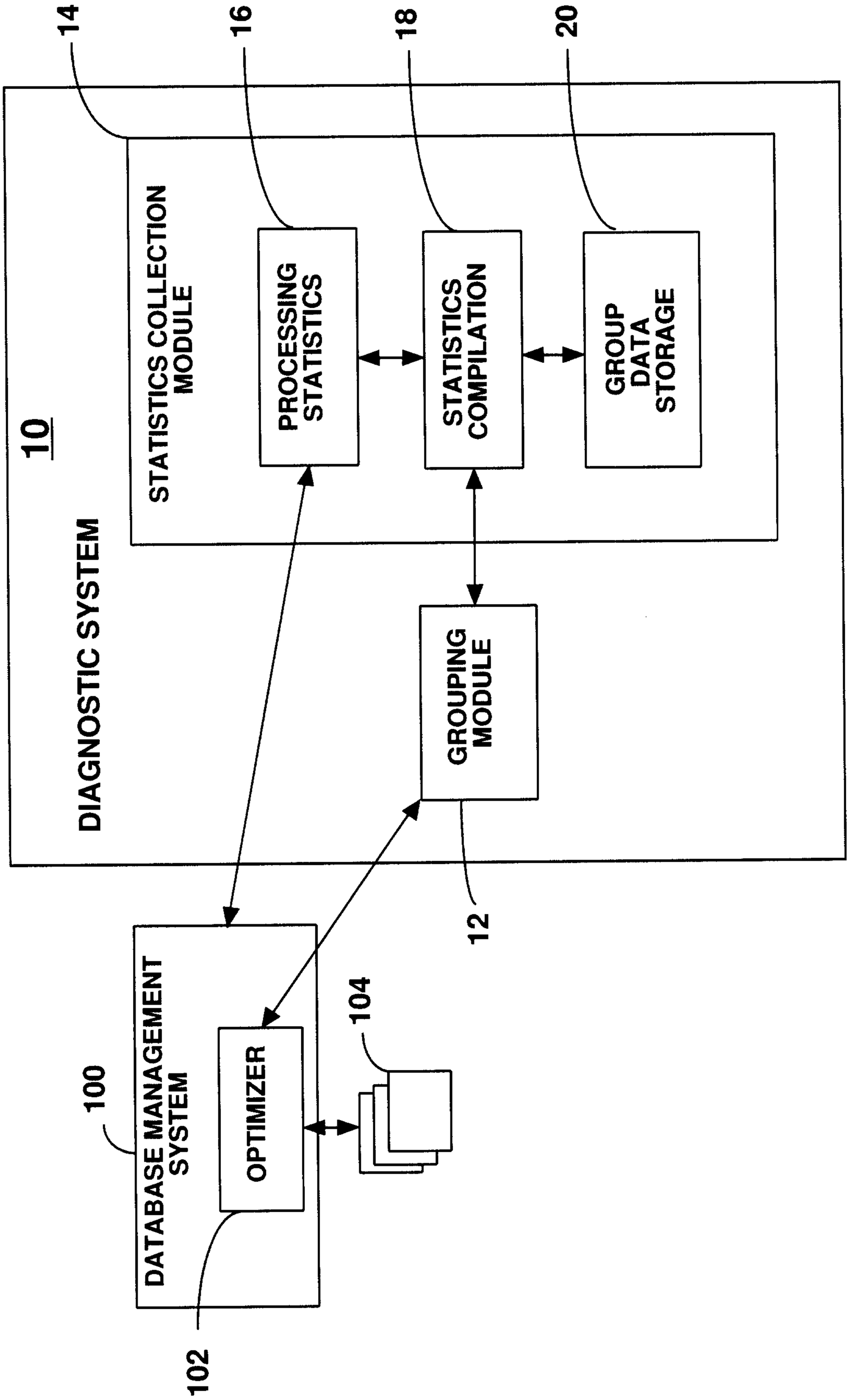

14
16
18
20
10
DIAGNOSTIC SYSTEM
STATISTICS COLLECTION MODULE
PROCESSING STATISTICS
STATISTICS COMPILATION
GROUP DATA STORAGE
GROUPING MODULE
12
100
DATABASE MANAGEMENT SYSTEM
OPTIMIZER
104
102

FIG. 1

**SAMPLE SQL QUERIES**

**(Q1) SELECT COUNT(*) FROM CARS WHERE MODEL='mustang' AND COLOUR='red';**

**(Q2) SELECT COUNT(*) FROM CARS WHERE MODEL='viper' AND COLOUR='silver';**

**(Q3) SELECT OWNER_NAME FROM CARS WHERE LICENSE_PLATE='ACTC403';**

**(Q4) SELECT COUNT(*) FROM CARS;**

**FIG. 2A**

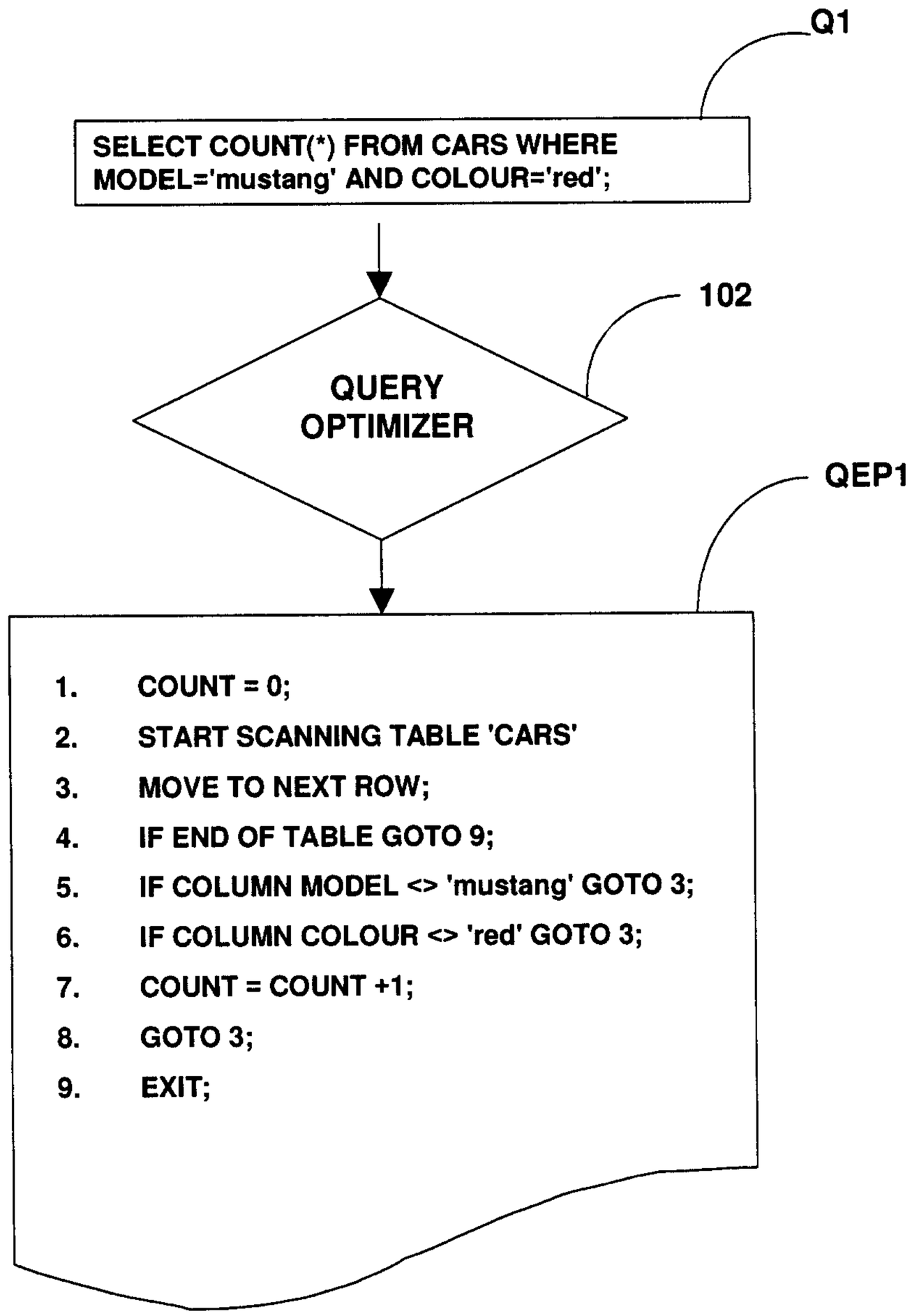

Q1
SELECT COUNT(*) FROM CARS WHERE
MODEL='mustang' AND COLOUR='red';
102
QUERY
OPTIMIZER
QEP1
1. COUNT = 0;
2. START SCANNING TABLE 'CARS'
3. MOVE TO NEXT ROW;
4. IF END OF TABLE GOTO 9;
5. IF COLUMN MODEL <> 'mustang' GOTO 3;
6. IF COLUMN COLOUR <> 'red' GOTO 3;
7. COUNT = COUNT +1;
8. GOTO 3;
9. EXIT;

FIG. 2B

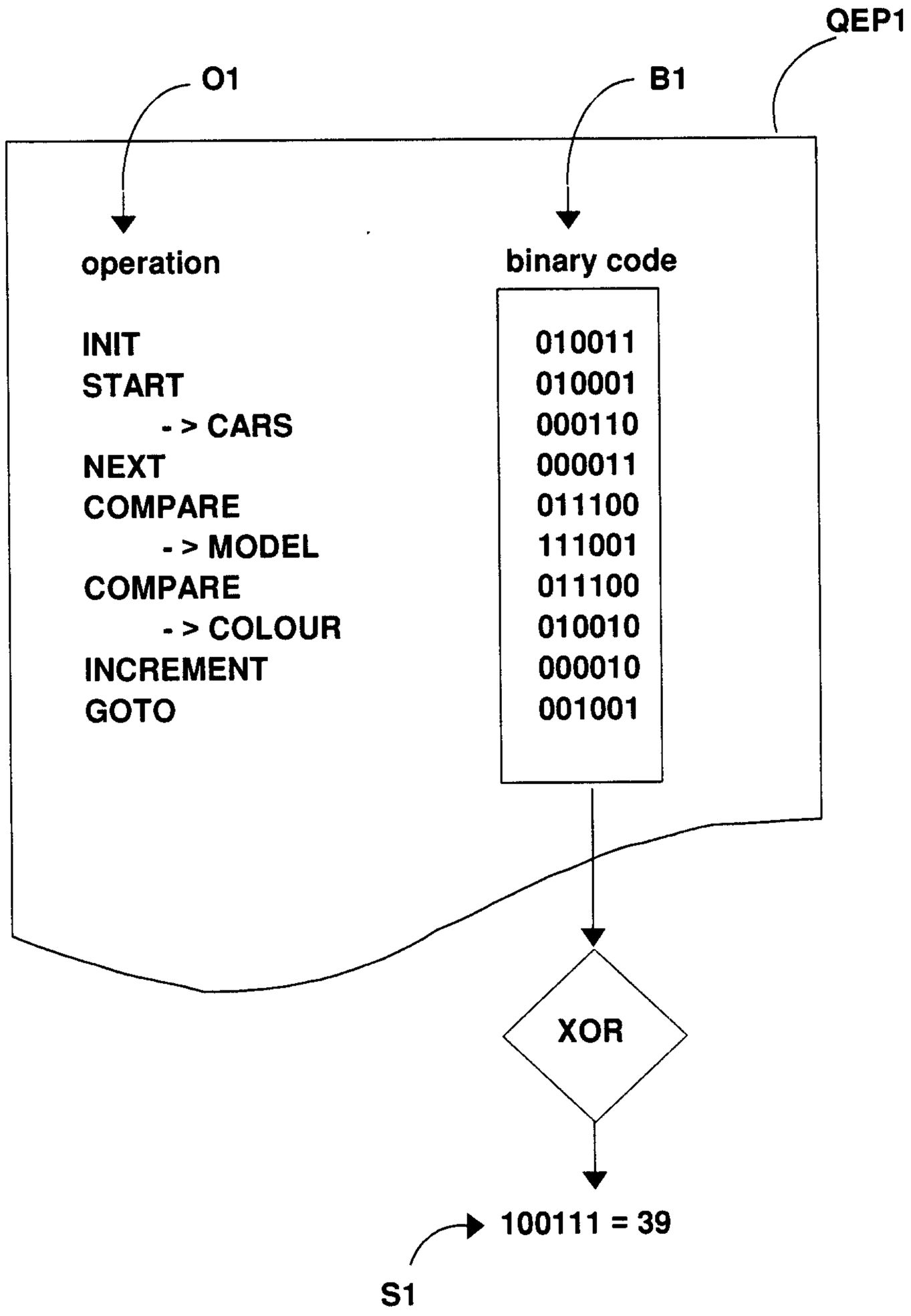

QEP1
O1
B1
operation
binary code
INIT
START
- > CARS
NEXT
COMPARE
- > MODEL
COMPARE
- > COLOUR
INCREMENT
GOTO
010011
010001
000110
000011
011100
111001
011100
010010
000010
001001
XOR
100111 = 39
S1

FIG. 2C

20

220

GROUP DATA STORAGE

| 202 Group ID | 204 Sample SQL Query | 206 No. of Queries | 208 CPU Time (sec) | 210 Total Sort Time (sec) | 212 Reads | 214 Writes |
|---|---|---|---|---|---|---|
| 39 | SELECT COUNT(*) FROM CARS WHERE MODEL='mustang' AND COLOUR='red' | 5000 | 9348 | 0 | 65663 | 0 |
| 19 | SELECT OWNER_NAME FROM CARS WHERE LICENSE_PLATE= 'ACTC403' | 3000 | 20 | 15 | 92 | 0 |
| 2 | SELECT COUNT(*) FROM CARS | 1 | 47 | 33 | 3 | 34 |
| ... | ... | ... | ... | ... | ... | ... |

FIG. 2D

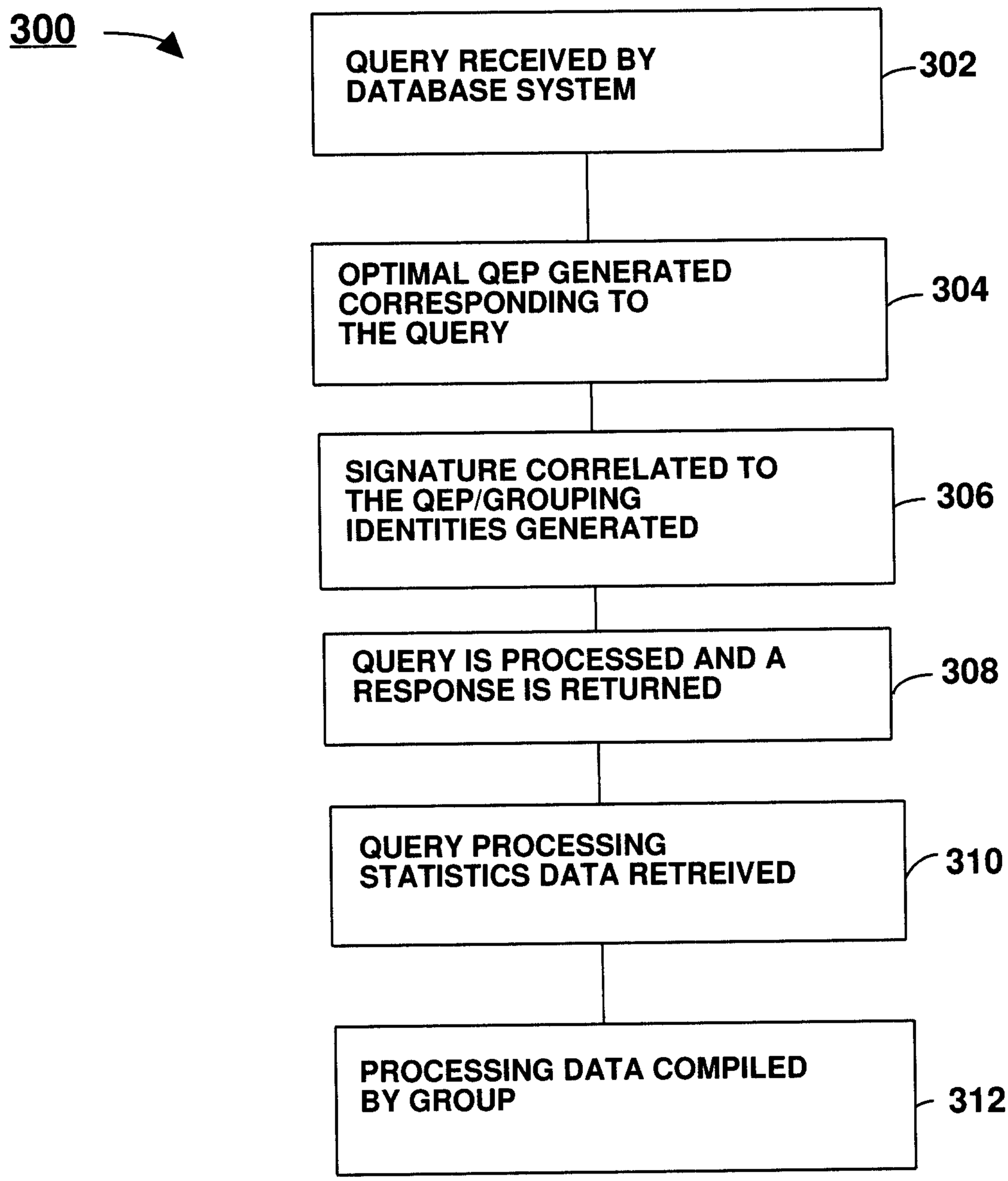
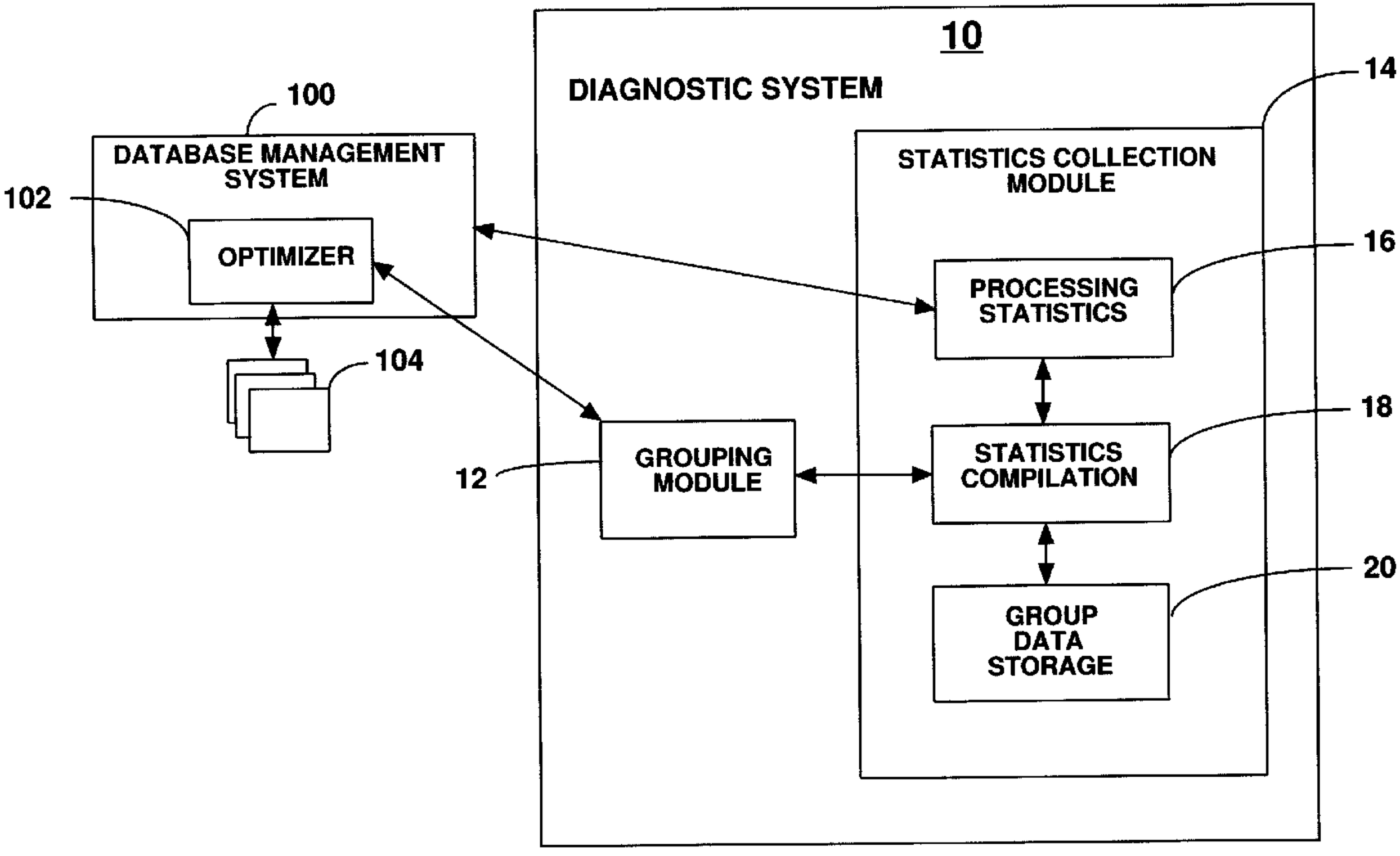

300
QUERY RECEIVED BY DATABASE SYSTEM
302
OPTIMAL QEP GENERATED CORRESPONDING TO THE QUERY
304
SIGNATURE CORRELATED TO THE QEP/GROUPING IDENTITIES GENERATED
306
QUERY IS PROCESSED AND A RESPONSE IS RETURNED
308
QUERY PROCESSING STATISTICS DATA RETREIVED
310
PROCESSING DATA COMPILED BY GROUP
312

FIG. 3


10
DIAGNOSTIC SYSTEM
14
100
DATABASE MANAGEMENT SYSTEM
102
OPTIMIZER
STATISTICS COLLECTION MODULE
16
PROCESSING STATISTICS
104
18
12
GROUPING MODULE
STATISTICS COMPILATION
20
GROUP DATA STORAGE