EXHIBIT A

26/ 210 #4/

PATENT

10-26-96

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application : Kenneth P. Baclawski
Serial No. : 08/318,252
Filed : October 5, 1994
For : DISTRIBUTED COMPUTER DATABASE SYSTEM AND METHOD
Examiner : C. Lewis
Attorney's Docket : NU-360XX

Group Art Unit: 2307

* *

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: *BOX NON-FEE AMENDMENT, Assistant Commissioner for Patents*, Washington, D.C. 20231 on 6/7/96.

By
Stanley M. Schurgin
Registration No. 20,979
Attorney for Applicant(s)

* *

AMENDMENT

BOX NON-FEE AMENDMENT
Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

In response to the Office Action dated April 16, 1996, please amend the above-identified patent application as follows:

Serial No.: 08/318,252
Filed: October 5, 1994
Group Art Unit: 2307

In the claims

Please amend claims 1, 6, 8, 12, 13 and 17 as follows:

Sub B1 1. (Amended) A method for information retrieval using fuzzy queries [of accessing data] in a distributed database system having a plurality of home nodes [node] and a plurality of query nodes connected by a network, said method comprising the steps of:

randomly selecting a first one of said plurality of home nodes;

fragmenting, by said selected home node, a query from a user into a plurality of query fragments;

hashing, by said selected home node, each said query fragment of said plurality of query fragments, said hashed query fragment having a first portion and a second portion;

transmitting, by said selected home node, each said hashed query fragment of said plurality of query fragments to a respective one of said plurality of query nodes indicated by said first portion of each said hashed query fragment;

using, by said query node, said second portion of said respective hashed query fragment to access data according to a local hash table located on said query node; and

returning, by each said query node accessing data according to said respective hashed query fragment, an object identifier corresponding to said accessed data to said selected home node.

Sub B2

6. (Amended) A method of storing data in a manner which is conducive to information retrieval using fuzzy queries in a distributed database system having a plurality of home nodes [node] and a plurality of query nodes connected by a network, said method comprising the steps of:

C2

randomly selecting a first one of said plurality of home nodes;

fragmenting, by said selected home node, data from a user into a plurality of data fragments;

hashing, by said selected home node, each said data fragment of said plurality of data fragments, said hashed data fragment having a first portion and a second portion;

transmitting, by said selected home node, each said hashed data fragment of said plurality of data fragments to a respective one of said plurality of query nodes indicated by said first portion of each said hashed data fragment; and

using, by said *query node*, said second portion of said respective hashed data fragment to store data according to a local hash table located on said query node.

Sub B4

C3

8. (Amended) A distributed database system having an information retrieval tool for handling queries from a user, comprising:

a plurality of home nodes [node]; and

a plurality of query nodes;

said plurality of home nodes [node] and said plurality of query nodes connected by a network,

3

wherein each said home node, upon receiving a query from a user, fragments said query into a plurality of query fragments, hashes each said query fragment of said plurality of query fragments into a hashed query fragment having a first portion and a second portion, and transmits each said hashed query fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed query fragment, and

further wherein each said query node [,] uses said second portion of said hashed query fragment to access data according to a local hash table located on said query node and returns [,] an object identifier corresponding to said accessed data to said home ~~node.~~

Sub B5

4

12. (Amended) A distributed database system for storage and retrieval of information, comprising:

a plurality of home nodes [node]; and

a plurality of query nodes;

said plurality of home nodes [node] and said plurality of query nodes connected by a network,

wherein each said home node, upon receiving data from a user, fragments said data into a plurality of data fragments, hashes each said data fragment of said plurality of data fragments into a hashed data fragment having a first portion and a second portion, and transmits each said hashed data fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed data fragment, and

wherein each said query node [,] uses said second portion of said hashed data fragment to store data according to a local hash table located on said query node.

P4

13. (Amended) A distributed database system having an information retrieval tool for handling queries from a user, comprising:

a plurality of home [node] nodes; and

a plurality of query nodes, said plurality of home nodes and said plurality of query nodes connected by a network,

each said home node, upon receiving a command from a user, enqueueing a predetermined task in response to said command,

a query task enqueued being resultant in, in response to a query command from said user, fragmenting a query contained in said query command into a plurality of query fragments, hashing each said query fragment of said plurality of query fragments into a hashed query fragment having a first portion and a second portion, and transmitting a query message containing each said hashed query fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed query fragment, said query node, upon receipt of said query message, using said second portion of said hashed query fragment to access data according to a local hash table located on said query node and transmitting a message returning an object identifier corresponding to said accessed data to said home node.

Sub B6

15

17. (Amended) A distributed database system for storage and retrieval of information, comprising:

a plurality of home node nodes; and

a plurality of query nodes, said plurality of home nodes and said plurality of query nodes connected by a network,

each said home node, upon receiving a command from a user, enqueueing a predetermined task in response to said command,

an insert task enqueued, in response to an insert command from said user, fragmenting data contained in said insert command into a plurality of data fragments, hashing each said data fragment of said plurality of data fragments into a hashed data fragment having a first portion and a second portion, and transmitting an insert message containing each said hashed data fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed data fragment,

said query node, upon receipt of said insert message, using said second portion of said hashed data fragment to store data according to a local hash table located on said query node.

REMARKS

The above-identified patent application has been amended and reconsideration is respectfully requested. Claims 1-17 are pending and stand rejected. Claims 1, 6, 8, 12, 13 and 17 have been amended.

Claims 1-17 are rejected under 35 U.S.C. §103 as being unpatentable over Chaturvedi, et al., "Scheduling the Allocation of Data Fragments in a Distributed Database Environment: A Machine Learning Approach", IEEE Transactions on Engineering Management, Vol. 41, No. 2, May 1994 and Houtsma et al., "Parallel Hierarchical Evaluation of Transitive Closure Queries", IEEE, 1991. With respect to claims 1, 6, 8, 12-13, and 17, the Examiner states that, "It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the hashing means of Houtsma's teachings with the teachings of Chaturvedi because the hashing means could enable Chaturvedi's information retrieval means to provide the queried node with a query value and a query identifier during the query nodes hashing process" (Paper No. 3, page 3). However, such a combination would not provide the distributed database and method of the present invention.

Both the Chaturvedi and Houtsma references describe techniques for partitioning files in a Distributed Relational Database System. These two references, and each of the papers cited by these two references, are in the field of **relational** database systems. A relational database system consists of one or more relations, also known as tables or files. Each relation is a set of records, also known as rows or tuples. Each record in a relation has a set of attributes, also known as fields or columns. Every record in a relation has exactly the same number of fields and the fields have the same types. For example, a customer relation might consist of a 40 character name field, a 60 character address field and a 6 digit customer identifier.

A fundamental characteristic of relational databases is that records **do not have object identity**. More particularly, each record is uniquely determined by the values of its fields. By contrast, data models other than the relational model generally assume that the basic objects do have object identity, i.e., an object exists independently of any attribute values it might have, and changing the attribute values will not change the object identity.

Another fundamental characteristic of relational databases is the use of a relational query language called the relational algebra. The relational algebra is roughly equivalent to what mathematicians call the "first order predicate calculus," and is primarily used for extracting information from a relational database system. However, the relational algebra may also be used for other purposes. For example, relational algebra expressions can be used to specify database views, security and authentication conditions, integrity constraints and database partitions. This *can be confusing, and has apparently* caused confusion in the examination of the present application, since these other uses of the relational algebra have nothing to do with extracting information from the database, and yet the word "query" is frequently used in connection with these other uses.

Modern relational databases typically deal with very large relations, i.e., relations that contain several terabytes (million megabytes) of data are common. The need to deal with such large relations along with the reduction in cost of computing equipment has driven the development of distributed relational database systems. A distributed relational database system is a relational database system that is distributed among a collection of computers which are connected by a communication network. Very large relations are distributed among the computers in the network by partitioning or otherwise breaking up the relations into disjoint pieces known as "fragments." These fragments are themselves relations, and typically contain in excess of tens or even hundreds of megabytes, even though the fragments are much smaller than the larger relation of which they are parts. Significantly, these relational fragments are **disjoint**.

The fragments of a distributed relational database system are *defined by using the relational algebra*. Perhaps as a result, the term "fragment query" is often used to refer to the relational algebra expression that defines a relational database fragment. This can be confusing, and has apparently caused confusion in the examination of the present application, since the relational algebra expression "fragment query" **does not describe extraction of information**, but rather provides the defining condition for the fragment.

The present invention does not utilize the relational model, and in particular does not utilize the relational models of Chaturvedi and Houtsma. A primary purpose of the present invention is to allow **information retrieval** for information objects that are more general than the simple records of a relational model system. For example, documents such as papers, books, World Wide Web pages, annotated images, and other documents can all be indexed using a search engine in accordance with the present invention. Significantly, none of these documents would be considered searchable records according to the relational model.

The present invention and the relational model express queries and records differently. The query language used by the present invention is the same language used to express the information objects, or more precisely their content labels, that are indexed by the search engine of the present invention (claims 1, 6, 8, 12, 13 and 17). This has the advantage that no additional language is required for expressing queries. In contrast, relational database system queries are expressed in the relational algebra and the records are expressed in other ways. The result of a query provided to the search engine of the present invention is a set of **object identifiers** (claim 1, line 17) with **weights** (claim 3, lines 2-3, claim 9) attached thereto. The weight attached to an object identifier represents ambiguous and fragmentary queries, which are also known as "fuzzy" queries. There is no analogous concept in the relational algebra. A relational algebra expression is a precise and unambiguous specification of a set of records. Using colloquial language, there is no "fuzziness" in the relational algebra.

The fragmentation technique of the present invention is different from fragmentation in the relational model. The present invention introduces a fragmentation technique that is utilized in the indexing algorithm. Information objects, or more precisely their content labels, are broken up into a collection of small overlapping fragments (claim 1, lines 4-5). The size of each fragment may typically be around 20 bytes. By contrast, the fragments of the relational model never overlap, are millions of times larger, and have a structure that is both conceptually and practically different. Furthermore, the present invention fragments both queries and information objects in the same way. This is impossible for relational model database systems, since queries and records have different structures.

With regard to the comment on Page 2, heading 3, sentence 1, Charturvedi introduces a new algorithm for defining fragments in a partitioned, distributed relational database system. As noted above, these relational fragments are unrelated to the object fragments of the present invention. This difference is illustrated by the cited example which uses the value of an attribute (named c) to break apart a base table (named T1) into two relation fragments, according to whether the attribute has value 'A' or 'B'.

With regard to the comment on Page 2, heading 3, sentence 2, Chaturvedi introduces a variation on the well known semijoin algorithm for computing a join. The join is one of the operators *of the relational algebra, and computing it efficiently is* important in relational database systems. Significantly, the algorithm for the two-way join described in Chaturvedi is very different from the algorithm used by the present invention. The Chaturvedi join query is split into two single-table sub-queries and then provided to the two nodes containing the base tables specified in the sub-queries. This splitting technique is commonly employed in Distributed Relational Database Systems. It is an algebraic factoring of the relational algebra expression that is the query. Algebraic factoring is a technique unrelated to the fragmentation of the present invention. More particularly, in the *present invention each fragment is hashed in its entirety (claim 1,* lines 6-8), and the hash value is provided to a node determined by the hash value itself. In the splitting technique in Chaturvedi, sub-queries are not hashed at all; they are shipped to the node containing the base table specified in the sub-query. This is hardly surprising as it would not make any sense to hash a relational query because the resulting hash value would not have any uses.

With regard to the comments associated with Figs. 2 and 3 of Chaturvedi, the architecture of Chaturvedi shown in those Figs. is quite different from the architecture of the present invention. More particularly, there is no central server in the present invention, and neither the nodes of the network nor the object fragments in the index have any kind of hierarchical structure. In the present invention the home node of a query is randomly chosen, and different queries will generally have different home nodes.

With regard to the comment on Page 2, heading 3, sentence 3, the database fragmentation mentioned by Chaturvedi in the Abstract is relational fragmentation and is unrelated to the fragmentation of the present invention. The fragment queries in Chaturvedi's Illustrative Examples (Page 198) are not query fragments, but rather relational algebra expressions used to define relation fragments. Numbers 1-4 in Example 2 on page 198 are queries that are in the query history at Site A. They are queries that at some time in the past were processed at Site A. They are used by the MLTIF to compute relational algebra expressions for defining relation fragments that would be better suited for evaluating the queries in the history than the current relation fragments. The presumption is that the past history is a good indicator of what the future will be. The MLTIF is **not a query evaluation algorithm** but rather a dynamic method for choosing good relation fragments in a Distributed Relational Database System. Therefore, the cited passages of the Chaturvedi reference are irrelevant to the present invention.

With regard to the comment on Page 3, heading 3, sentence 1, nowhere on Page 197, column 1 or Page 199, column 2 of Chaturvedi is there any mention of a local hash table or any hashing operation.

With regard to the comment on Page 3, heading 3, sentence 2, no object identifiers are mentioned on page 198 of Chaturvedi. Indeed, since the relational model explicitly rejects object identity, it would be amazing if it did mention object identifiers. The Illustrative Example on page 198 simply discusses how to find relational algebra expressions for defining time invariant relational fragments.

With regard to the comment on Page 3, heading 3, sentence 3, no hashing operation is mentioned anywhere in the Abstract.

With regard to the comment on Page 3, heading 3, sentence 4, Houtsma does not teach use of hashing. Indeed, on page 130, column 2, par. 3, Houtsma refers to a number of papers that use different methods to solve the transitive closure problem, including hash-based methods. Houtsma teaches a disconnection set approach that does not use hashing. Further, the graph shown in Houtsma is an auxiliary structure used in the algorithm. The graph defines a notion of adjacency between relation fragments. This is unrelated to the graphs (semantic networks) used in the present invention. As discussed above, the fragments of the present invention are quite different from the fragments of the relational model. Since the fragments of the present invention are parts of the semantic network, there is no concept of fragment adjacency in the present invention. In Houtsma, the graph has the relation fragments as the vertices, with unlabeled edges defined by relation fragment adjacency, while in the present invention the fragments may be regarded as fragments of a graph having labeled edges (semantic relationships) that connect concept instantiations with one another.

With regard to the comment on Page 3, heading 3, sentence 5, no hashing operation is mentioned here or anywhere in the reference. The fragment H is the high speed fragment. The term "high speed" was probably chosen because of their motivating example: the railway network of many European countries. It could equally well have been called the "special fragment" or the "wide-connection fragment."

The Examiner has also rejected claims 2, 7 and 14 based on the Chaturvedi and Houtsma references. However, Houtsma does not use hashing and Chaturvedi does not solve the **information retrieval** problem of the present invention. The Chaturvedi network architecture is very different from the architecture of the present invention. In Chaturvedi, except for the central server node, it is presumed that the servers are located where the queries will be presented by users. By contrast, the architecture of the present invention is a search engine that is entirely remote from any user nodes. The "home node" in Chaturvedi is the user node itself, i.e., the node where the query is presented to the distributed system. The "home node" in the present invention is one of the nodes in the search engine, and it can be randomly chosen by one of the front end processors. Further, Chaturvedi never fragments a query.

In addition to the architectural differences, there are no concepts of measure of relevance or degree of relevance (claims 3, 9) in the relational model, and no such concepts are mentioned or employed in Chaturvedi. In particular, the use of the word "relevance" in Chaturvedi is unrelated to the "fuzzy" notion of relevance in the present invention. Like all research on relational systems, Chaturvedi employs no notion of weighted relevance. When it is stated, for example, that "...it [join-value set] is transmitted to the relevant nodes participating in the join operation," Chaturvedi simply means that the join-value set is sent to those nodes participating in the join which may contribute any tuples to the result of the join. There is no relevance weighting involved in this operation. If it can be determined that a node participating in the join will not contribute any tuples to the result, then it is not sent the join-value set, otherwise the join-value set is sent to the node. The decision is completely "sharp" and does not involve any "fuzziness." This is hardly surprising since Chaturvedi describes a relational model which is unrelated to information retrieval using fuzzy queries.

With regard to fragment storage, the storage of relation fragments in a Distributed Relational Database System is specified in the allocation schema. In Chaturvedi, Example 4, there are three relation fragments: T1A, T1B, and T2. T1A is the relation fragment defined by the relational algebra expression:

```
SELECT * FROM T1 WHERE e = 'A'
```

and T1B is the relation fragment defined by the relational algebra expression:

```
SELECT * FROM T1 WHERE e = 'B'
```

The allocation schema simply specifies which nodes contain a copy of each relation fragment. Here, for example, is the allocation schema used by Chaturvedi in this example:

T1A: node A

T1B: node B

T2: node A

It is merely coincidence that the value of the attribute e coincides with the name of the node.

With regard to the comments on Page 4, the steps in Chaturvedi, page 199, column 1 are concerned with choosing time invariant relation queries. These steps are not concerned with query processing per se. In the present invention a query request can specify one of several levels of service (claim 16). Roughly speaking, the lower levels of service are faster but are less accurate, the higher levels of service are slower but more accurate. This notion of level of service is meaningless for the relational model. In the relational model, all queries have **exactly one correct answer.** There is no concept in the relational model of answers that are better or worse.

In sum, the field of "information retrieval using fuzzy queries" (a term of art) is quite different from the relational model. In the relational model a query is a complete and unambiguous specification of the result. Relevance in the relational model is either TRUE or FALSE. In information retrieval results are returned which may or may not satisfy the intentions behind the query, and which may even be unrelated to the intentions behind the query. The claims have been amended to particularly point out this difference and remove the confusion which has apparently been brought about by the use of terms which are similar to those of the cited references.

For the reasons given above, reconsideration and allowance is respectfully requested. The Examiner is encouraged to telephone the undersigned attorney to discuss any matters in furtherance of the prosecution of this application.

Respectfully submitted,

Kenneth P. Baclawski

By
Stanley M. Schurgin
Registration No. 20,979
Attorney for Applicant(s)

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES
Ten Post Office Square
Boston, Massachusetts 02109

Telephone: (617) 542-2290
Telecopier: (617) 451-0313

Date: 6/7/96

SMS/jet
84277

#6/B
W. Lawson
1-7-97

PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application	:	Kenneth P. Baclawski
Application No.	:	08/318,252
Filed	:	October 5, 1994
For	:	DISTRIBUTED COMPUTER DATABASE SYSTEM AND METHOD
Examiner	:	C. Lewis
Attorney's Docket	:	NU-360XX

Group Art Unit: 2307

* *

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: BOX NON-FEE AMENDMENT, Assistant Commissioner for Patents, Washington, D.C. 20231 on 12/11/96.

By
Stanley M. Schurgin
Registration No. 20,979
Attorney for Applicant

* *

AMENDMENT

BOX NON-FEE AMENDMENT
Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

In response to the Office Action dated September 11, 1996, please amend the above-identified patent application as follows:

In the Claims

Please amend claims 1, 6, 7, 8, 12, 13 and 17 as follows:

1. (Twice Amended) A method for information retrieval using fuzzy queries in a non-relational, distributed database system having a plurality of home nodes and a plurality of query nodes connected by a network, said method comprising the steps of:

randomly selecting a first one of said plurality of home nodes;

fragmenting, by said selected home node, a query from a user into a plurality of query fragments;

hashing, by said selected home node, each said query fragment of said plurality of query fragments, said hashed query fragment having a first portion and a second portion;

transmitting, by said selected home node, each said hashed query fragment of said plurality of query fragments to a respective one of said plurality of query nodes indicated by said first portion of each said hashed query fragment;

using, by said query node, said second portion of said respective hashed query fragment to access data according to a local hash table located on said query node; and

returning, by each said query node accessing data according to said respective hashed query fragment, an object identifier corresponding to said accessed data to said selected home node.

6. (Twice Amended) A method of storing [data] objects in a manner which is conducive to information retrieval using fuzzy queries in a non-relational, distributed database system having a plurality of home nodes and a plurality of query nodes connected by a network, said method comprising the steps of:

randomly selecting a first one of said plurality of home nodes;

fragmenting, by said selected home node, [data] objects from a user into a plurality of [data] object fragments;

hashing, by said selected home node, each said [data] object fragment of said plurality of [data] object fragments, said hashed [data] object fragment having a first portion and a second portion;

B2

transmitting, by said selected home node, each said hashed [data] object fragment of said plurality of data fragments to a respective one of said plurality of query nodes indicated by said first portion of each said hashed [data] object fragment; and

using, by said query node, said second portion of said respective hashed [data] object fragment to store data according to a local hash table located on said query node.

B3

7. (Amended) The method of claim 6 further comprising the step of receiving, at said home node, said [data] objects from said user, prior to the step of fragmenting said [data] object.

8. (Twice Amended) A non-relational, distributed database system having an information retrieval tool for handling queries from a user, comprising:

a plurality of home nodes; and

a plurality of query nodes;

said plurality of home nodes and said plurality of query nodes connected by a network,

45 wherein each said home node, upon receiving a query from a user, fragments said query into a plurality of query fragments, hashes each said query fragment of said plurality of query fragments into a hashed query fragment having a first portion and a second portion, and transmits each said hashed query fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed query fragment, and

further wherein each said query node uses said second portion of said hashed query fragment to access data according to a local hash table located on said query node and returns an object identifier corresponding to said accessed data to said home node.

12. (Twice Amended) A non-relational, distributed database system for storage and retrieval of information objects, comprising:

65 a plurality of home nodes; and

a plurality of query nodes;

said plurality of home nodes and said plurality of query nodes connected by a network,

wherein each said home node, upon receiving [data] an object from a user, fragments said [data] object into a plurality of [data] object fragments, hashes each said [data] object fragment of said plurality of [data] object fragments into a hashed [data] object fragment having a first portion and a second portion, and transmits each said hashed [data] object fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed [data] object fragment, and

wherein each said query node uses said second portion of said hashed [data] object fragment to store [data] objects according to a local hash table located on said query node.

13. (Twice Amended) A non-relational, distributed database system having an information retrieval tool for handling queries from a user, comprising:

a plurality of home nodes; and

a plurality of query nodes, said plurality of home nodes and said plurality of query nodes connected by a network,

each said home node, upon receiving a command from a user, enqueueing a predetermined task in response to said command,

a query task enqueued being resultant in, in response to a query command from said user, fragmenting a query contained in said query command into a plurality of query fragments, hashing each said query fragment of said plurality of query fragments into a hashed query fragment having a first portion and a second portion, and transmitting a query message containing each said hashed query fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed query fragment,

said query node, upon receipt of said query message, using said second portion of said hashed query fragment to access data according to a local hash table located on said query node and transmitting a message returning an object identifier corresponding to said accessed data to said home node.

---

17. (Twice Amended) A non-relational, distributed database system for storage and retrieval of information, comprising:

a plurality of home node nodes; and

a plurality of query nodes, said plurality of home nodes and said plurality of query nodes connected by a network,

each said home node, upon receiving a command from a user, enqueueing a predetermined task in response to said command,

an insert task enqueued, in response to an insert command from said user, fragmenting data contained in said insert command into a plurality of data fragments, hashing each said data fragment of said plurality of data fragments into a hashed data fragment having a first portion and a second portion, and transmitting an insert message containing each said hashed data fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed data fragment,

. said query node, upon receipt of said insert message, using said second portion of said hashed data fragment to store data according to a local hash table located on said query node.

REMARKS

Claims 1-17 are pending in this application. Claims 1, 6, 8, 12, 13 and 17 have been amended.

The Examiner has rejected claims 1-17 under 35 U.S.C. § 102(b) as being anticipated by Neches, U.S. Patent No. 5,006,978. Neches teaches breaking up and distributing a relational database with a hash function for facilitation of data storage. Claims 1-5, 8-11 and 13-17 of the present application do not relate to storage of data, and are therefore not suggested by Neches. Claims 6, 7 and 12 of the present application relate to storage of data. However, claims 6, 7 and 12 are distinguished from Neches since these claims (as amended) recite method and apparatus for fragmenting, hashing and distributing **objects**. Operating upon objects is significantly different from operating upon relations because of size, content and structural differences. For example, a relation will typically be much larger than an object, and will not include methods.

Relational database systems consist of relations, sometimes referred to as tables or files, where each such relations is a set of records, sometimes referred to as rows or tuples. Each record in such a relation has a set of attributes, also known as fields or columns. Significantly, each record in a relation has exactly the same number of fields, and the fields have the same types. For example, a customer relation could consist of a forty character name field, a sixty character address field and six digit customer identifier. Further, records in relational database systems do not have object identity. More particularly, each record is uniquely determined by the values of its fields.

The present invention expresses queries and records differently than the relational databases of Neches and the previously cited references. The query language used by the present invention is used to express information objects that are indexed by the search engine. In contrast, relational database queries are expressed in a relational algebra and records are expressed in other ways. The present invention therefore provides an advantage since separate "languages" are not required for expressing queries and records. Further, the result of a query provided to the search engine of the present invention is a set of object identifiers with weights attached thereto. Such results do not necessarily contain each term in the query or provide relevant information, and are therefore known as "fuzzy" queries. In contrast, relational algebra expressions return a precise and unambiguous set of records, each of which is relevant since it satisfies each term in the relational algebra, and hence there is no "fuzziness" in the relational database model. The claims therefore recite these distinguishing features.

For the reasons stated above it is suggested that claims 1-17 are allowable, and reconsideration and allowance are respectfully requested. The Examiner is invited to telephone the undersigned attorney to discuss any matters which would expedite allowance of present application.

Respectfully submitted,

KENNETH P. BACLAWSKI

By
Stanley M. Schurgin
Registration No. 20,979
Attorney for Applicant

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
Ten Post Office Square
Boston, Massachusetts 02109

Telephone: (617) 542-2290
Telecopier: (617) 451-0313

Date: 12/4/96

SMS/jet
93808

PATENT

6-3-97
#8

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application : Kenneth P. Baclawski
Application No. : 08/318,252
Filed : October 5, 1994
For : DISTRIBUTED COMPUTER DATABASE SYSTEM AND METHOD
Examiner : C. Lewis
Attorney's Docket : NU-360XX

Group Art Unit: 2307

* *

I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: BOX NON-FEE AMENDMENT, Assistant Commissioner for Patents, Washington, D.C. 20231 on 5/14/97.

By Stanley M. Schurgin
Stanley M. Schurgin
Registration No. 20,979
Attorney for Applicant

* *

AMENDMENT

BOX NON-FEE AMENDMENT
Assistant Commissioner for Patents
Washington, D.C. 20231

Sir:

In response to the Office Action dated March 21, 1997, reconsideration is respectfully requested in view of the following remarks:

REMARKS

Claims 1-17 are pending in this application. Applicant is pleased to acknowledge allowance of claims 3-5, 9-11 and 14-16. Claims 1, 2, 6-8, 12, 13 and 17 have been rejected in view of Kuechler. However, the present invention as claimed is patentably distinct from Kuechler.

As described at various points throughout columns 1-20, Kuechler employs a single node system for storing and manipulating information. At column 20, lines 60-68 and column 21, lines 1-30 Kuechler discusses a distributed version of the disclosed method. However, even in this distributed version Kuechler *only describes* employing the same node as the home node. Hence, Kuechler makes no distinction between a home node and a query node as recited in each of the independent claims of the present invention.

In addition to failing to distinguish home nodes from query nodes, Kuechler broadcasts the same query to every processing node *(column 21, lines 9-10). Hence, the query is not fragmented* as recited in the claims of the present invention. Further, the information elements (i.e., records) are distributed by storing whole records on the processing nodes, and these information elements are also not fragmented. The location of an information element is determined by its record number, not by any *information* contained in the record. By contrast, the present invention describes a fundamentally distributed technique, and both queries and objects are fragmented. In the present invention query fragments are processed only on the node for which the query fragment is relevant, query fragments are not broadcast to all the nodes, objects are fragmented, and the information content of an object fragment is used to determine on which node it is to be stored. Further, objects are not stored on a single node. Because objects are fragmented and because these fragments are stored independently, objects are distributed over many nodes. These distinguishing features are recited in the claims and hence distinguish the present invention from Kuechler.

The Kuechler concept of a query is the one used by the relational model. Such a query is unambiguous in the sense that every record either satisfies the query or it does not. There is no "fuzziness." The Kuechler query processing technique does introduce additional records that may or may not satisfy the query, but this is done for the sake of improving performance, not because there is any fuzziness in the query. A final filtering step (Fig.1 item 32) removes the spurious records. By contrast, the present invention employs an intrinsically "fuzzy" notion of query. Objects satisfy the query to a greater or lessor degree. Higher levels of service in the present invention are designed to improve the estimates of the degrees by which objects satisfy the query rather than to eliminate spurious objects. Such higher levels of service are optional, whereas the final filtering step of Kuechler is mandatory. *Furthermore, the distribution* of processing effort for the higher levels of service in the present invention are very different from the distribution of processing effort for the final filtering step in Kuechler. Kuechler assigns compact symbols or codes (Abstract, line 7 and *column 8, lines 6-7) to ranges of* attribute values. These codes are assigned unique codes. They are very different from hash values, which are computed, not assigned, and which are not unique. Finally, Kuechler does not use any hashing techniques. *The topological maps of Kuechler are stored* using some form of bit map (column 17, lines 51-61) rather than using a hash table.

Application No.: 08/318,252
Filed: October 5, 1994
Group Art Unit: 2307

For the reasons stated above it is submitted that claims 1, 2, 6-8, 12, 13 and 17 are allowable, and reconsideration and allowance are respectfully requested. The Examiner is invited to telephone the undersigned attorney to discuss any matters which would expedite allowance of present application.

Respectfully submitted,

KENNETH P. BACLAWSKI

By
Stanley M. Schurgin
Registration No. 20,979
Attorney for Applicant

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
Ten Post Office Square
Boston, Massachusetts 02109

Telephone: (617) 542-2290
Telecopier: (617) 451-0313

Date: 5/19/97

SMS/rec
101555