The opinion in support of the decision being entered today was not written for publication and is not binding precedent of the Board.

Paper No. 23

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte DANIEL FRANCIS LIEUWEN, AKIRA KAWAGUCHI, INDERPAL SINGH MUMICK, DALLAN WENDELL QUASS and KENNETH A. ROSS

Appeal No. 2003-1062
Application No. 09/004,265

ON BRIEF

Before KRASS, RUGGIERO and BLANKENSHIP, Administrative Patent Judges.

KRASS, Administrative Patent Judge.

DECISION ON APPEAL

This is a decision on appeal from the final rejection of claims 1-28.

The invention pertains to database management systems. In particular, concurrency control in materialized views of a database is maintained. As explained in the principal brief,

sometimes, instead of running a query on the database itself, a part of that database will be copied to another location and the query will be run on that copy. The copy is known as a “view.” Concurrency becomes important to ensure that different parties, using different copies, will deal with consistent data, i.e., the “concurrency” of the data must be maintained.

In order to provide concurrency, the instant invention employs a “locking” technique, i.e., access is blocked to certain data so that only one party at a time can modify the locked data. When a process seeks to generate a “view,” that process identifies the data of interest within the database. That data is termed a “tuple.” The process requests a lock on the tuple. Responsive to this request, another process will lock not only the tuple, but also a “superset” of the tuple. A “superset” is a set which contains the tuple, but also contains additional data, or tuples.

Representative independent claim 1 is reproduced as follows:

1. A system, comprising:

a database;

means for generating views of the database, said views containing view tuples; and

means for detecting issuance of a read-lock-request for a target view tuple and, in response, locking tuples in the database which include a superset of tuples from which the target view tuple is derived.

The examiner relies on the following references:

Reiter et al. (Reiter) 5,666,526 Sep. 9, 1997

Colby et al., "Algorithms for deferred view maintenance", ACM International Conference on Management of Data and Symposium on Principles of Database Systems, June 3-6, 1996, pp. 469-480.

"Locking in a Binary Relational Data Data Base", IBM Technical Disclosure Bulletin, (IBM) vol. 25, No. 10, March 1983, pp. 5027-5028.

Claims 1-28 stand rejected under 35 U.S.C. § 103. As evidence of obviousness, the examiner offers IBM and Colby with regard to claims 1, 2, 4-16 and 18-28, adding Reiter with regard to claims 3 and 17.

Reference is made to the briefs and answer for the respective positions of appellants and the examiner.

OPINION

In rejecting claims under 35 U.S.C. § 103, the examiner bears the initial burden of presenting a prima facie case of obviousness. See In re Rijckaert, 9 F.3d 1531, 1532, 28 USPQ2d 1955, 1956 (Fed. Cir. 1993). To reach a conclusion of obviousness under § 103, the examiner must produce a factual

basis supported by a teaching in a prior art reference or shown to be common knowledge of unquestionable demonstration. Our reviewing court requires this evidence in order to establish a prima facie case. In re Piasecki, 745 F.2d 1468, 1471-72, 223 USPQ 785, 787-88 (Fed. Cir. 1984). The examiner must not only identify the elements in the prior art or that knowledge generally available to one of ordinary skill in the art would lead the individual to combine the relevant teachings of the references. In re Fine, 837 F.2d 1071, 1074, 5 USPQ2d 1596, 1598 (Fed. Cir. 1988).

It is the examiner's position that IBM discloses a database with triples (tuples). The examiner notes that IBM does not specifically teach "views" of a database, but the examiner relies on Colby for a teaching of a materialized view of databases associated with data warehouses (Colby, p. 469, abstract), concluding that it would have been obvious "to apply Colby to IBM, because of Colby's taught advantage of database views, providing a user of IBM a way to visualize results from users concurrently accessing said triples" (answer-page 4).

The examiner also notes that IBM does not specifically teach a "read-lock-request" but that this would have been obvious "because IBM's teaching of database locks for deadlock prevention

suggests both read and write locks, which are both common and known tools for preserving database data integrity (IBM near bottom; compare with claim 1 "*means for detecting issuance of a read-lock-request for a target view tuple*")" (answer-page 4).

With regard to the claim limitation "in response...," the examiner alleges "in response to a lock request of a record, a collection (superset) of tuples factored into a number of sub collections are locked, in addition to locking said record (IBM at middle; compare with claim 1 "*in response, locking tuples in the database which include a superset of tuples from which a target view tuple is derived*")" (answer-page 4).

We will sustain the rejection of claims 1, 2, 4-16 and 18-28 under 35 U.S.C. § 103 because, in our view, the examiner has set forth a prima facie case of obviousness which has not been successfully rebutted by appellants.

Initially, we note that it is our view that Colby is merely cumulative to what is already suggested by IBM. That is, while the examiner relies on Colby for a teaching of "views" of a database, IBM already implicitly discloses such "views." Appellants have defined a "view" as being a part of a database which is copied to another location and appellants have defined a "tuple" as being data of interest within the database (presumably

within the view) (See pages 2-3 of the principal brief). Since IBM is concerned with "triples,"[1] and it is clear that triples, or tuples, are merely data of interest in a "view," there is clearly an implied generation of "views" of the database within IBM, and these "views" contain view tuples, or view triples.

Having said that, we find it unnecessary to address appellants' argument that the references are not combinable since Colby, albeit cumulative to IBM, is not even necessary for the rejection, in our view.

Appellants argue that the references do not disclose the claimed locking of a "superset of tuples" since IBM explicitly states that "a superset of the necessary triples is held by the accessing application, while only a subset of the necessary triples is explicitly locked." Accordingly, appellants conclude that a "superset of tuples" is not locked in IBM.

We disagree. A "superset" is merely "a set which contains the tuple, but additional data, or tuples" (principal brief-page 3). IBM goes on to explain that a triple, or tuple, is defined in general by the form (x, y, z), where x, y, and z are identifiers of database elements which name entities in the real

[1]It is clear that IBM's "triples" are identical to appellants' "tuples" Appellants do not dispute this.

world, giving, as an example, triples which represent record structures, where each triple takes the form (key, relationship name, attribute field). In discussing triple locking, IBM states that this is achieved by passing to a lock manager only the x component of a triple, and that this "effectively locks all triples of the form (x, ?, ?), since locking requests for any triples of this form cannot be distinguished." Thus, it is clear to us that in locking the triple, or tuple, IBM also locks all triples belonging to the set of triples of the form (x, ?, ?). Since the triple of interest, (x, y, z) is derived from the set of triples (x, ?, ?), i.e., (x, ?, ?) is the "superset" of triples, or tuples, from which the target view triple (x, y, z) is derived, the instant claim language is met.

Appellants argue that the set (x, ?, ?) is not a "superset" of tuples from which the target view tuple is derived but that the set (x, ?, ?) merely describes other tuples which are part of the same set to which the target view tuple (x, y, z) belongs but that these other tuples are not a "superset" of the target view tuple. Appellants give the example of being in a crowd in a building with other people, contending that a man in that crowd may be locked in the building with other people but the other people do not comprise a "superset" of the man. The analogy is

not persuasive to us. The other people in the crowd, by themselves, may not constitute a “superset” as to the man, just as the other tuples, by themselves, may not constitute a “superset” relative to the target view tuple (x, y, z). However, the man in the crowd is part of the entire crowd and the entire crowd (including the man) is a “superset” of the man, just as the target view tuple (x, y, z) is part of the set (x, ?, ?) and the entire set (including (x, y, z)) would be a “superset” of the target view tuple (x, y, z), under appellants’ definition of a set containing “the tuple, but additional data, or tuples”.

Appellants also argue that there is nothing in the references showing that the “locking” is “in response to” detection of a “read-lock-request.” In particular, appellants attempt to distinguish between locks, themselves, which are clearly described in IBM, and a “request” for a lock. Appellants state that the existence, or suggestion, of locks in the reference does not imply the existence of ”requests for locks.” We disagree.

If a data processing system performs a locking function, artisans would have understood that there had to be a “request” for that locking function, either by a user, or by the processing program itself, or both. A processor does not perform functions,

e.g., locking, unless it is told (i.e., “requested”) to do so, either by manual intervention by a user or via its programming. It appears that appellants are reading the term “request” too narrowly.

In any event, there does not appear to be a need for such “interpretations” as to whether a locking “request” is suggested by IBM because the IBM reference, itself, clearly indicates that there are “locking requests” (see the disclosure 9 and 12 lines up from the bottom of the page of IBM). Manifestly, if there is a locking request, as disclosed by IBM, there must inherently be a way for the system to “detect” that request in order to carry out the request. Accordingly, there is also some means for detecting issuance of a read-lock-request, as claimed, in IBM.

Appellants also argue that the read-lock-request “applies to something in the **view**, not in the **database**” (principal brief-page 7). We are unsure as to what exact claim language appellants rely on for this point since no claim language appears to make this requirement, nor are we sure why the disclosure of IBM would not fit this alleged situation. Appellants appear to be saying that the means for detecting is in the view while the thing being locked is in the database but we find no such language in the instant claims. Accordingly, appellants’ argument in this regard

is not persuasive.

With regard to claim 2, appellants argue that IBM does not show the claimed “wherein the superset is limited to a derivation set of the target view tuple.” We disagree. As explained supra, the superset in IBM includes those tuples of the form (x, ?, ?). It is clear that the target view tuple (x, y, z) is a derivation of the broader (x, ?, ?) tuples constituting the “superset.”

With regard to claim 4, appellants argue that this claim requires a transaction T to issue the read-lock-request and that prior to the issuance of the request, the transaction T declares a read set which contains all data to be read by transaction T. It is appellants’ view that the applied references do not suggest this limitation. We disagree. As explained by the examiner, at pages 4-5 of the answer, IBM discloses a transaction which, subsequent to a record locking attempt, reads a set of related sub collection of triples (x, ?, ?) to see if any triples are already locked, and if none are locked, a lock request is issued to complete the record locking attempt.

Appellants argue, at page 25 of the reply brief, that the examiner may have shown that an independent transaction has occurred but that claim 4 requires the “read set.” The “read

set" in IBM would appear to be the (x, ?, ?) triples.

With regard to claim 5, appellants argue that the instant claimed invention requires that the transaction T locks a subset of the read set **BEFORE** processing certain operations (page 27 of the principal brief). We do not find anything different in the IBM disclosure and appellants have not pointed to anything contrary in the IBM reference.

With regard to claim 6, appellants argue that the examiner has not shown that a "subset" contains "all base data members of the read set." It appears to us that a subset (x, y, z) would contain all base members of the read set (x, ?, ?), where x is a "base member." Appellants have not demonstrated anything to the contrary. Merely pointing out what a claim recites and alleging that the applied references do not disclose this limitation fails to show error on the part of the examiner's rationale.

With regard to claim 7, appellants argue that Colby does not show the "maintenance" claimed because the claim requires the maintenance to be performed on a "set of views in the read set." Again, appellants merely recite claim language and assert that it is not shown in the references. However, the examiner has explained how/why the references are combined and appellants have

pointed to no error in the examiner's reasoning. For example, appellants contend that the claim requires the maintenance to be performed on a "set of views in the read set" but does not show why the IBM/Colby combination is not deemed to perform the maintenance on a set of views in the read set.

Claims 8-10 fall with claims 1, 4 and 7 in accordance with appellants' statement at page 29 of the principal brief.

Independent claims 11 and 12 comprise limitations similar to those discussed supra and the rejection of these claims is sustained for the reasons supra.

With regard to claims 13 and 14, appellants argue that these claims require that a transaction is aborted if a "condition" occurs, namely that a warning is issued, and that the warning is only issued under certain circumstances. Appellants assert that the mere existence of the process of aborting in IBM is insufficient to show these conditional relationships (principal brief-page 33). The "warning" in IBM would be the disclosure that "if any subcollection of triples is already locked, the locking request is refused and locks already acquired are freed" (i.e., the lock is aborted, as claimed).

Claims 15, 16 and 18-24 contain limitations which have been discussed supra and, for those reasons, we will also sustain the

rejection of these claims under 35 U.S.C. § 103.

With regard to claims 3 and 17, the examiner adds Reiter to IBM and Colby. Reiter is used to show a “dependency graph,” as recited in claim 3. Appellants argue that that dependency graph is not used to “ascertain” the members of the “superset” and that the examiner has not shown that this “ascertainment” is of dependency “between the view containing the target view tuple and the base data.”

We agree with appellants as to the limitations of claims 3 and 17. While Reiter does, indeed, disclose a “dependency graph,” we find no reason for the artisan to have combined this teaching of Reiter with anything in IBM to result in ascertaining members of a superset by the use of a dependency graph between the view containing the target view tuple and the base data.

The rationale for the combination stated by the examiner, <u>to wit</u>, “because of Reiter’s taught advantage of a dependency graphs (sic), providing IBM a way to ascertain related tuple collections,” is unconvincing. We find no evidence that would have led the artisan to conclude that there was any advantage in applying a dependency graph in IBM nor is it clear to us why or how the artisan would have applied such a dependency graph in

IBM.

CONCLUSION

We have sustained the rejection of claims 1, 2, 4-16 and 18-28 under 35 U.S.C. § 103 but we have not sustained the rejection of claims 3 and 17 under 35 U.S.C. § 103.

Accordingly, the examiner's decision is affirmed-in-part.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 CFR § 1.136(a).

AFFIRMED-IN-PART

ERROL A. KRASS Administrative Patent Judge	)	
JOSEPH F. RUGGIERO Administrative Patent Judge	)	BOARD OF PATENT APPEALS AND INTERFERENCES
HOWARD B. BLANKENSHIP Administrative Patent Judge	)	

EK/RWK

S.H. DWORETSKY
AT&T BELL LABORATORIES
600 MOUNTAIN AVENUE
P O BOX 636
MURRAY HILL, NJ 07974-0636