Trials@uspto.gov
571-272-7822

Paper 7
Entered: May 14, 2015

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

ACTIFIO, INC.,
Petitioner,

v.

DELPHIX CORP.,
Patent Owner.

Case IPR2015-00100
Patent 8,566,361 B2

Before KARL D. EASTHOM, JENNIFER S. BISK, and PATRICK R. SCANLON, *Administrative Patent Judges.*

SCANLON, *Administrative Patent Judge.*

DECISION
Institution of *Inter Partes* Review
37 C.F.R. § 42.108

I. INTRODUCTION

Petitioner, Actifio, Inc., filed a Petition (Paper 1, "Pet.") to institute an *inter partes* review of claims 1–6, 8, 14, 16–19, 24, and 25 of U.S. Patent No. 8,566,361 (Ex. 1001, "the '361 patent") pursuant to 35 U.S.C. §§ 311–319. Patent Owner, Delphix Corp., filed a Preliminary Response (Paper 6, "Prelim. Resp.").

We have jurisdiction under 35 U.S.C. § 314, which provides that *inter partes* review may not be instituted unless "the information presented in the petition . . . and any [preliminary] response . . . shows that there is a reasonable likelihood that the petitioner would prevail with respect to at least 1 of the claims challenged in the petition." 35 U.S.C. § 314(a).

Upon consideration of the Petition and the Preliminary Response, and for the reasons discussed below, we determine that Petitioner has established a reasonable likelihood of prevailing in showing the unpatentability of the challenged claims. Accordingly, we institute *inter partes* review on claims 1–6, 8, 14, 16–19, 24, and 25.

II. BACKGROUND

A. Related Matters

The parties indicate that the '361 patent is at issue in *Delphix Corp. v. Actifio, Inc.*, Case No. 5:13-cv-04613-BLF (N.D. Cal.). Pet. 2; Paper 4, 2. The '361 patent is also the subject of another petition for *inter partes* review filed by Petitioner—IPR2015-00108. Pet. 2; Paper 4, 2. In addition, Petitioner has filed petitions seeking *inter partes* review of other patents owned by Patent Owner disclosing similar subject matter—IPR2015-00014, IPR2015-00016, IPR2015-00019, IPR2015-00025, IPR2015-00026, IPR2015-

00034, IPR2015-00050, IPR2015-00052, IPR2015-00128, and IPR2015-00136. Pet. 2.

B. The '361 Patent (Ex. 1001)

The '361 patent describes systems and methods for managing databases and lifecycle workflows based on databases. Ex. 1001, 1:12–14. More specifically, the '361 patent involves creating one or more virtual databases based on a production database or another virtual database at a particular point in time. *Id.* at 5:64–66. Virtual databases are created "using storage level snapshots of production databases or clones of production databases instead of a live production database." *Id.* at 6:16–19. "A virtual database created for a point in time is stored as a set of files that contain the information of the database as available at that point in time." *Id.* at 6:31–34.

Figure 1 of the '361 patent is reproduced below.

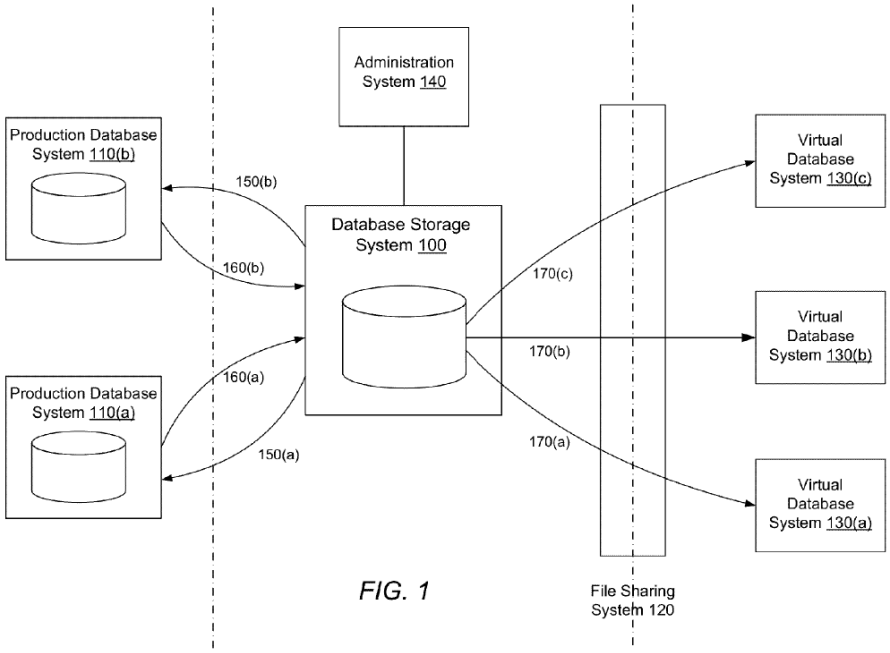

FIG. 1

Figure 1 illustrates how information may be copied from production database systems 110 to database storage system 100. *Id.* at 7:12–15. To create a

virtual database, database storage system 100 creates files representing the corresponding information from a production database system at a given point in time. *Id.* at 7:50–52. More particularly, database storage system 100 retrieves and stores information from production database systems 110. *Id.* at 9:38–39. Database storage system 100 then exposes the files to virtual database system 130 using file sharing system 120. *Id.* at 7:53–55. Virtual database system 130 includes database server 230 and operating system 240. *Id.* at 7:60–8:6, Fig 2(a).

C. Illustrative Claim

Of the challenged claims in the '361 patent, claims 1, 14, 17, and 24 are independent. Claim 1 is illustrative of the claims at issue:

> 1. A method for replicating a database, the method comprising:
>
> linking a source database, wherein linking the source database comprises receiving information identifying the source database;
>
> loading the source database at multiple points in time, wherein the loading comprises:
>
> receiving database blocks for different point-in-time copies of the source database, and
>
> storing the database blocks in a first storage system;
>
> replicating the database blocks of the source database from the first storage system to a second storage system; and
>
> provisioning a virtual database (VDB) from the second storage system to a system running a database server, wherein provisioning comprises:
>
> creating a set of files linked to the stored database blocks on the second storage system, and

> mounting the set of files to the system allowing the database server running on the system to access the set of files.

Ex. 1001, 35:39–58.

D. The Prior Art

Petitioner relies on the following prior art:

1. JAWAHAR LAL, ROGER SANDERS & JEREMY BRUMER, DB2: CLONING A DATABASE USING NETAPP FLEXCLONE™ TECHNOLOGY, TR-3460 (2006) ("Sanders") (Ex. 1004);

2. John K. Edwards et al., *FlexVol: Flexible, Efficient File Volume Virtualization in WAFL*, PROC. OF THE 2008 USENIX ANNUAL TECHNICAL CONF. 129 (2008) ("Edwards") (Ex. 1005); and

3. DARRIN CHAPMAN, MIKE FEDERWISCH, & CHUCK DUFRESNE, SNAPMIRROR® BEST PRACTICES GUIDE, TR-3446 (2006) ("Chapman") (Ex. 1006).

E. The Asserted Ground

Petitioner challenges claims 1–6, 8, 14, 16–19, 24, and 25 of the '361 patent as obvious under 35 U.S.C. § 103 over Sanders, Edwards, and Chapman. Pet. 3.

III. ANALYSIS

A. Claim Construction

We interpret claims of an unexpired patent using the broadest reasonable construction in light of the specification of the patent in which they appear. *See* 37 C.F.R. § 42.100(b); *In re Cuozzo Speed Techs., LLC.*, 778 F.3d 1271, 1281 (Fed. Cir. 2015). On this record and for purposes of this Decision, we determine that only the claim terms addressed below require express construction.

1. "database"

All the disputed claim terms include some form of the term "database." The parties do not provide expressly a claim construction for the term. The '361 patent states that "[a] database comprises data stored in a computer for use by computer implemented applications." Ex. 1001, 6:1–2. On this record, under the broadest reasonable construction, a database comprises data stored for use by a computer.

2. "database block"

All the challenged independent claims—1, 14, 17, and 24—recite "database blocks." The claim language itself provides some context from which to determine at least a partial meaning for this term. For example, the claims recite receiving "database blocks for different point-in-time copies of the source database." From this language, we can ascertain that "database blocks" are at least components of a database.

a. The Parties Proposed Definitions

Petitioner basically argues that the understanding gleaned from the claim language is the broadest reasonable interpretation of the term. Specifically, Petitioner proposes that "database block" means "a unit of data used by a database." Pet. 9 (citing Ex. 1016 ¶¶ 107) (emphasis omitted). To support its position, Petitioner points to the '361 patent's statement that "a virtual database 'includes a set of database blocks and the data structures for referring to the database blocks.'" *Id.* at 8–9 (quoting Ex. 1001, 6:31–35).

In its Preliminary Response, Patent Owner argues that the term "database block" as used in the '361 patent should be construed more narrowly to mean "a unit of data used by a database which comprises a

specific number of bytes stored in the storage, a portion of which stores metadata associated with the block." Prelim. Resp. 28 (emphasis omitted).

b. The '808 Patent

Patent Owner asserts that U.S. Patent No. 8,150,808 B2 ("the '808 patent"), which Patent Owner describes as being related to the '361 patent, explicitly defines the term "database block." Prelim. Resp. 28. Specifically, Patent Owner points to the following language (*id*. at 29):

> A database block is a unit of data used by a database and comprises a specific number of bytes stored in the storage. A database block can also be referred to as a page. **A portion of the database block stores metadata associated with the database block**. Examples of information that may be stored in the metadata of a database block include information related to the data stored in the database block, information related to objects of database that the database block is part of, or information indicating when the data in the database block was updated.

Ex. 2002, 2:7–17 (emphasis added by Patent Owner).

We are not persuaded that this language is an explicit definition of "database block" as opposed to a description of a "database block" as implemented in a specific embodiment. The first portion of this passage is consistent with the language in the claims themselves, as described above—"a database block is a unit of data used by a database." Ex. 2002, 2:7–9. The subsequent language, however, adds features beyond those described in the claims, including that the "database block" comprises "a specific number of bytes stored in the storage," and "stores metadata." *Id.* at 2:9–12. It is not clear that these additional features are required components of a "database block" as opposed to optional features employed in specific situations. For example, the very next sentence in this passage lists "[e]xamples of

information that *may be stored* in the metadata." *Id.* at 2:12–17 (emphasis added).

Thus, for purposes of this Decision, we do not consider the passage of the '808 patent quoted above to be an explicit definition of the term "database block."

c. Metadata Associated with the Database Block

Patent Owner also asserts that a person of ordinary skill in the art would have understood the term "database block" to include metadata. Prelim. Resp. 29–30. As evidence of this understanding, Patent Owner points to documentation describing the Oracle database system, which discloses a database block that stores information about the table and row to which the data stored in the block belongs. *Id.* at 30 (citing Ex. 2003, 71, 117). Patent Owner also points to language in the '361 patent stating that "[t]he point-in-time copy manager 310 analyzes 520 the metadata for each database block to determine if the database block needs to be stored in the storage system data store 390 or it can be eliminated." Prelim. Resp. 30–31 (quoting Ex. 1001, 14:43–46). According to Patent Owner, in a particular embodiment of the '361 patent, "[m]etadata is the key to identifying the subset of database blocks that should be stored which, as discussed above, is one of the advantages of the Delphix technology." *Id.* at 32.

Although we agree with Patent Owner that "database blocks" *can* include metadata associated with the block, we are not persuaded, on this record, that "database blocks" *require* that such metadata is stored with the database block. Patent Owner points to documentation describing Oracle databases, arguing that Oracle is "one of the database systems with which the invention may operate." *Id.* at 8–9. The '361 patent, however, indicates that

the disclosed system may employ any of the commercially available database systems, such as "ORACLE, SYBASE, MICROSOFT SQL SERVER, IBM's DB2, MYSQL, and the like." Ex. 1001, 6:5–9. Patent Owner does not argue all of these commercially available database systems require metadata as part of their database blocks. In fact, Patent Owner contends that the NetApp system, which includes a WAFL (write anywhere file layout) system based on an IBM DB2 system, does *not* include metadata in a database block. *See* Prelim. Resp. 34. Hence, the evidence of record shows that the features of the Oracle database system described by Patent Owner are, at best, features relating to a particular embodiment described in the '361 patent, which may not be read into the claims "absent clear disclaimer in the specification." *In re Am. Acad. of Sci. Tech Ctr.*, 367 F.3d 1359, 1369 (Fed. Cir. 2004).

On this record, we are not persuaded that the broadest reasonable construction of the term "database block" requires a portion storing metadata associated with the block.

d. A Specific Number of Bytes Stored in the Storage

Patent Owner appears to argue that the portion of its proposed definition stating "a specific number of bytes stored in the storage" means that a "database block" must not be the same size as an operating system block. Prelim. Resp. 32–33. Patent Owner, relying on Oracle documentation, describes an "operating system block" as "the minimum unit of data that the operating system can read or write." *Id.* at 32 (citing Ex. 2004, 250). According to Patent Owner, an "Oracle block," on the other hand, "is a logical storage structure whose size and structure are not known to the operating system." *Id.* (citing Ex. 2004, 250). Thus, according to Patent Owner,

"operating system blocks are distinct from the data blocks used by the Oracle database system." *Id.* at 33.

It is unclear that the phrase "a specific number of bytes stored in the storage," actually provides the meaning that Patent Owner appears to ascribe to the phrase. In fact, it is unclear why, even if we adopted this portion of Patent Owner's proposed definition, the "specific number of bytes" of a "database block" could not be equivalent to the number of bytes in an operating system block. Further, even assuming as true Patent Owner's definitions of "database block" and "operating system block," it is unclear why a "database block" could not be stored as *one* "operating system block." In other words, nothing in Patent Owner's proposed definition distinguishes the size of a "database block" from an "operating system block," so Patent Owner's arguments on this specific point appear to be irrelevant. Moreover, even if the phrase—"a specific number of bytes stored in the storage,"—could be read as Patent Owner proposes, we are not persuaded that the '361 patent limits the term "database block" as distinct from an operating system block.

On this record, we are not persuaded that that the broadest reasonable construction of the term "database blocks" requires a specific number of bytes stored in the storage.

e. Database Block v. Data Block

Finally, Patent Owner argues that a database block must be something more than just "any 'unit' of data" (Prelim. Resp. 32), adding "[h]ad Delphix intended to claim a broader term, such as a generic 'data block,' it could have used those words instead. It did not" (*id*. at 33). In our reading of the '361 patent, however, it does appear that the terms "database blocks" and "blocks of data" are used interchangeably. For example, the '361 patent states "[t]he

database blocks retrieved by a point in time copy manager 310 . . . can be used to reconstruct a copy of a database in the production system 110." Ex. 1001, 11:7–11 (emphasis added). In the very next paragraph, the '361 patent continues this discussion by stating that "the point-in-time copy manager 310 may call APIs of storage allocation manager to save *blocks of data* retrieved from the production database system 110." *Id.* at 11:15–18 (emphasis added). Thus, we are not persuaded that Patent Owner intended the term "database block" to be more limited than the unquestionably broad term "block of data."

f. Conclusion

For purposes of this Decision, consistent with the language of the claims and the Specification, the broadest reasonable construction of a "database block" is "a unit of data used by a database."

3. "virtual database"

All the challenged independent claims 1, 14, 17, and 24 recite the term "virtual database." The claim language itself provides some context from which to determine at least a partial meaning for this term. For example, the claims recite provisioning a virtual database to a system. Provisioning the virtual database comprises creating a set of files linked to stored database blocks and mounting that set of files to the system.

a. The Parties Proposed Definitions

Petitioner proposes that "virtual database" means "a set of database files capable of being read from and written to, created by pointing to already-stored database blocks." Pet. 8 (citing Ex. 1016 ¶¶ 102) (emphasis omitted). To support its position, Petitioner asserts the '361 patent "discloses a method for creating a virtual database that involves creating files by creating a file structure that includes pointers to the database blocks of a point-in-time copy

of a source database," and "the claims and the specification [of the '361 patent] make clear that the set of files in a virtual database must be capable of being 'read from and written to.'" *Id.* 7–8 (citing Ex. 1001, 6:19–22; Ex. 1016 ¶¶ 98, 100).

In its Preliminary Response, Patent Owner argues that the term "virtual database" as used in the '361 patent should be construed to mean "a set of files to which a database server can read and write such that the physical implementation of the database files is decoupled from the logical use of the database files by the database server." Prelim. Resp. 39 (citing Ex. 1001, 6:19–22) (emphasis omitted).

b. The Database Server

One difference between Petitioner's and Patent Owner's proposed constructions of the term "virtual database" is that Patent Owner includes the term "database server" in its construction and Petitioner does not. We are not persuaded that inserting "database server" into the definition of "virtual database" is necessary to give meaning to the term. *See Renishaw PLC v. Marposs Societa' per Azioni*, 158 F.3d 1243, 1249 (Fed. Cir. 1998); *E.I. du Pont de Nemours & Co. v. Phillips Petroleum Co*., 849 F.2d 1430, 1433 (Fed. Cir. 1988). Figure 3 of the '361 patent shows a "virtual database *system*," with a "database server" and a "VDB system library." Ex. 1001, Fig. 3. Also, under the heading "Virtual Database Systems," the '361 patent states that "[a] database comprises data . . . for use by computer implemented applications" and "[a] database server is a computer program that can interact with the database and provides database services, for example, access to the data stored in the database." *Id.* at 6:1–5. This language implies that a database server interacts with a database, but is not a *part* of the database. Thus, the

disclosure of the ’361 patent does *not* show that a virtual database requires a database server or that the meaning of a virtual database cannot be understood without a database server.

Therefore, the record does not support inserting “database server” into the definition of “virtual database.” *See also*, IPR2015-00016, Paper 11, 12–17 (construing the term “virtual database” as used in the ’808 patent not to include “database server”).

c. “the physical implementation of the database files is decoupled from the logical use of the database files”

Patent Owner relies (Prelim. Resp. 39) on the following sentence in the ’361 patent to support the “decoupled” aspect of its construction: “The virtual databases are ‘virtual’ in the sense that the physical implementation of the database files is decoupled from the logical use of the database files by a database server.” Ex. 1001, 6:19–22. The ’361 patent does not describe expressly what it means for “the physical implementation of the database files [to be] decoupled from the logical use of the database files by a database server.”

Patent Owner contends that this decoupling is equivalent to how the virtualization process is generally used in computing technologies, and is exemplified by the Java programming language running on a Java Virtual Machine (JVM). Prelim. Resp. 39–40 (citing Ex. 2003, 451–52, Fig. 24-7). According to Patent Owner, the Java Virtual Machine sits between a Java application and the host operating system interpreting the Java instructions and translating them to be understandable by whatever specific platform it is running on. *Id.* at 40. Patent Owner analogizes this description of the Java Virtual Machine to a “virtual database” by asserting that “the creation and use

of virtual databases does not depend on, and is not tied to, the underlying storage system of the source database system." *Id.* at 41–42. However, the '361 patent describes a Java Virtual Machine as part of database *system* 110 (Ex. 1001, 9:13–15), which is separate from virtual database 220 stored on database storage system 100. Hence, Patent Owner's discussion of the Java Virtual Machine may relate to database servers or database applications running on a database system, but it is *not* directed to a virtual database stored on a database storage system.

Similarly, Patent Owner's argument regarding virtualization specific to the '361 patent is directed to the database *system*, not the virtual database. For example, Patent Owner argues "each database (130c)" depicted in Figure 1 of the '361 patent is "virtual" (Prelim. Resp. 41) when, in fact, Figure 1 shows Virtual Database *System* 130(c), which is *not* the virtual database residing on Database Storage System 100. Hence, these arguments by Patent Owner do not shed light on what is meant by "the physical implementation of the database files [] decoupled from the logical use of the database files," as the arguments are *not* directed to virtual databases.

Instead, we look directly to the '361 patent for guidance. Figure 12 of the '361 patent is reproduced below.

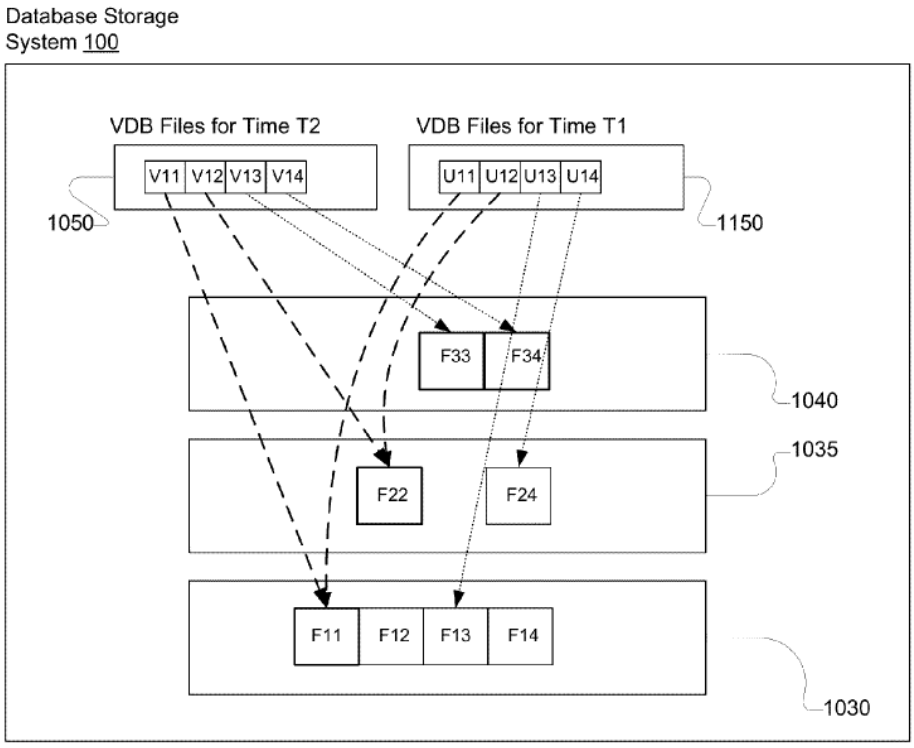

FIG. 12

Figure 12 shows how database blocks may be shared by file structures created for several virtual databases. Ex. 1001, 19:63–65. As shown in the Figure, file structures 1050 and 1150 include blocks V11–V14 and U11–U14 respectively. Several blocks in the virtual database may share the same database blocks. For example, V11 and U11 both point to the same copy of F11. *Id.* at 20:4–5. Thus, the '361 patent indicates that the database files are "virtual" in the sense that they create the illusion of storing data, but the system actually stores the data elsewhere in another physical location (F11) specified by the pointing blocks such as V11 and U11. The sharing of blocks across multiple VDBs provides efficient utilization of data stored in the storage system data store. *Id*. at 20:7–9.

Based on this disclosure, "the logical use of database files" can be understood as reading from or writing to database blocks by reading from or

writing to virtual database file structures that *point* to actual database blocks instead of actually containing data. Hence, as described in the '361 patent, decoupling the physical implementation of the database files from the reading or writing to database blocks, is accomplished by using pointers to map blocks in virtual database files to physical addresses for database blocks stored in physical storage devices. Therefore, for purposes of this Decision, we construe the term "virtual database" to be slightly different than either of the parties proposed constructions—a set of database files capable of being read from and written to, and capable of being mapped to physical addresses for stored database blocks.

B. Asserted Ground Based on Sanders, Edwards, and Chapman

Petitioner challenges claims 1–6, 8, 14, 16–19, 24, and 25 as obvious under 35 U.S.C. § 103(a) over Sanders, Edwards, and Chapman. Pet. 3, 28–59. To support this assertion, Petitioner relies on the Declaration of Dr. Erez Zadok (Ex. 1016).

1. Overview of Sanders

Sanders describes a system "to clone a DB2 database quickly and easily." Ex. 1004, 1. "Database cloning is [a] process by which you can create an exact copy of a DB2 database." *Id*. at 3. The disclosed NetApp system uses FlexClone and SnapMirror technologies in a combined manner. *Id*. at 8. This combined technology allows administrators to clone a production FlexVol database system as a writable FlexClone database on another storage system. *Id*. at 8–9. Specifically, FlexClone provides point-in-time copies of the production database. *Id.* at 3. Further, "[a] FlexClone volume is a writable point-in-time image of a FlexVol volume or another

FlexClone volume." *Id.* Stated differently, "[t]he clone database is a frozen image of the database file system at the time of the clone creation. If necessary, the primary database can be restored from the snaphsot created for the clone; or applications can point directly to the clone database." *Id.* at 6.

A FlexClone volume "uses space very efficiently, allowing both the original FlexVol volume and the FlexClone volume to share common data, storing only the data that changes between the original volume and the clone." *Id.* at 3. Clones can be created on the same or different storage systems. *Id.* at 6. The SnapMirror technology provides FlexClone volumes to be produced at different destinations: "A SnapMirror source and its corresponding destination can reside on the same storage system or on two separate storage systems that are miles apart." *Id.* at 3. Figure 3 of Sanders is reproduced below:

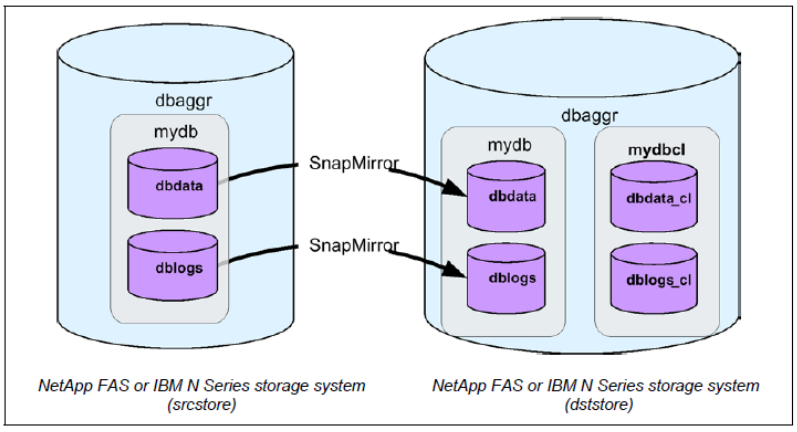

Figure 3) Clone database volumes on the second NetApp FAS or IBM N Series storage system

Figure 3 depicts using the combined SnapMirror and FlexClone technology to transmit point-in-time clone copies of a production database to a destination storage system. *See id.* at 8.

Sanders also describes mounting a clone database on a database server. *See id.* at 14 ("In order to access the clone database, you need to mount the clone volumes to a database server. . . . [Y]ou can mount the clone volume by

executing the following command on the database server: mount [MountPoint].").

2. Overview of Edwards

Edwards describes creating virtual instances of WAFL (write anywhere file layout) file systems; read-only versions, called "FlexVol" volumes (Ex. 1005, 009)[1] and writeable versions, called "FlexClone" volumes (*id.* at 014–015). A FlexVol volume "break[s] this tight bond between volumes and their underlying storage" by hiding a file system, which spans a pool of storage, and creating externally visible volumes inside the files of the hidden file system. *Id.* at 011. This "introduces a level of indirection, or virtualization, between the logical storage space used by a volume and the physical storage space." *Id.*

Edwards discloses that writable versions of the FlexVol volumes, FlexClones, are particularly useful in database environments—"for example, it is often desirable to make writable copies of a production database for development or test purposes." *Id.* at 014. To create a FlexClone volume, WAFL creates a copy of vol_info block of the Snapshot copy on which the clone is based. *Id.* at 015. The FlexClone volume, thus, inherits pointers to the complete file system stored in the original Snapshot copy. *Id.*

3. Overview of Chapman

Chapman describes NetApp's SnapMirror technology and is "intended to serve as a deployment guide for architecting and deploying SnapMirror in a customer environment." Ex. 1006, 1. As one of several features, Chapman discloses "cascading" as a "variation on the basic SnapMirror deployment and

[1] When referencing Exhibit 1005, we refer to the pagination inserted by Petitioner.

function involves mirroring from established mirrors to more SnapMirror destinations." *Id.* at 22.

4. Petitioner's Assertions

a. Independent claims 1 and 17

Regarding independent claim 1, Petitioner argues that "Sanders discloses a process for replicating a database." Pet. 28 (citing Ex 1004, 3; Ex. 1016 ¶ 114). In particular, Petitioner asserts that Sanders discloses receiving information identifying a source database—and, thus, linking the source database—because the reference discloses "creating a snapshot copy of a source volume containing an IBM DB2 database on a source storage system and then mirroring the copy from the source volume to a destination volume on a first storage system." *Id.* (citing Ex 1004, 22–23; Ex. 1016 ¶ 116).

According to Petitioner, Sanders discloses loading a source database at multiple points in time (*id.* (citing Ex. 1016 ¶ 118)) and Edwards discloses that the "data blocks" or "blocks" of Sanders are units of data used by a database (i.e., "database blocks") (*id.* at 31 (citing Ex. 1005, 021)). Petitioner then asserts that "Sanders in view of Edwards discloses receiving and storing database blocks for different point-in-time copies of a source database in a first storage system." *Id.*

Petitioner asserts that Sanders discloses using "NetApp's SnapMirror technology for data replication" and Chapman discloses "the use of data cascading, which is a 'variation on the basic SnapMirror deployment and function involves mirroring from established mirrors ***to more SnapMirror destinations***." *Id.* at 32 (citing Ex. 1004, 22, 27; Ex. 1006, 22; Ex. 1016 ¶ 126). Petitioner then asserts:

> Given Sanders's disclosure that a "SnapMirror source and its corresponding destination can reside on […] two separate storage systems that are miles apart" (Sanders at 3), a POSITA would have been motivated to incorporate Chapman's teachings that cascading "make[s] a uniform set of data available on a read-only basis to users from various locations throughout a network" (Chapman at 22) and can thus be "very useful for data distribution." *Id.* at 11. Thus, Sanders in view of Chapman discloses [the replicating database blocks] limitation. Zadok Decl. at ¶ 129.

Id. at 33.

Petitioner asserts that Sanders discloses making a "virtual database available to a database server running on a storage system through the use of FlexClone to create clones of databases on FlexVol volumes, followed by the mounting of those clones to a database server." *Id.* at 34 (citing Ex. 1004, 3, 28; Ex. 1016 ¶ 130). In addition, Petitioner argues that Chapman discloses that "any volume that resides on a cascaded, second destination storage system can be cloned." *Id.* (citing Ex. 1016 ¶ 131). Petitioner concludes the combination of Sanders and Chapman discloses the limitation of provisioning a virtual database. *Id.* at 35 (citing Ex. 1016 ¶ 133).

Petitioner argues that Sanders discloses creating a database clone that "is a frozen image of the database file system at the time of the clone creation." *Id.* (quoting Ex. 1004, 6 (emphasis omitted); citing Ex. 1016 ¶ 135). Petitioner also argues that Edwards discloses "further detail on creating a set of files linked to the stored database blocks on the storage system, through the creation of a FlexClone volume." *Id.* at 35–36 (citing Ex. 1016 ¶ 135). Petitioner contends that one of ordinary skill in the art "would have looked to Edwards to understand the specifics of how FlexClone creates files linked to stored database blocks on the storage system." *Id.* at

37–38 (citing Ex. 1016 ¶ 140). Finally, Petitioner relies on Sanders for disclosing mounting the set of files. *Id.* at 38 (citing Ex. 1004, 28, 29; Ex. 1016 ¶¶ 141, 142).

Independent claim 17 is directed to a "computer program product having a non-transitory computer-readable storage medium storing computer-executable code for replicating a database." Ex. 1001, 37:54–56. Petitioner notes that the computer executable code comprises instructions to "perform steps essentially identical to those recited in Claim 1." Pet. 55. Thus, Petitioner asserts that, "for the reasons discussed above in connection with Claim 1, Sanders, in view of Edwards and Chapman, discloses and renders obvious the computer program product elements of Claim 17." *Id.*

b. Claims 2 and 18

Claim 2 depends from claim 1 and further recites that "the replicating of database blocks comprises transmitting the database blocks from the first storage system to the second storage system for storing in the second storage system." Ex. 1001, 35:59–62. Similarly, claim 18 depends from claim 17 and further recites the same subject matter regarding the replicating of database blocks. *Id.* at 38:9–12. Petitioner contends that Chapman discloses the additionally recited limitations of claims 2 and 18. Pet. 38–39, 56 (citing Ex 1006, 10, 11; Ex. 1016 ¶¶ 144, 194).

c. Claim 3

Claim 3 depends from claim 1 and further recites that "the replicating of database blocks comprises transmitting a subset of database blocks comprising database blocks that changed since a point-in-time from the first storage system to the second storage system for storing in the second storage system." Ex. 1001, 35:63–67. Petitioner contends that Chapman discloses the

additional limitations of claim 3. Pet. 39–40 (citing Ex 1006, 10, 11; Ex. 1016 ¶¶ 145, 146).

d. Claim 4

Claim 4 depends from claim 1 and further recites using the virtual database from the second storage system as a standby database when the source database is unavailable. Ex. 1001, 36:1–4. Petitioner asserts that Sanders discloses this limitation. Pet. 40–41 (citing Ex 1004, 5; Ex. 1016 ¶¶ 147, 148).

e. Claims 5 and 19

Claim 5 depends from claim 1 and further recites "transmitting database blocks stored in the second storage system to the first storage system" and "storing the database blocks received from the second storage system at the first storage system." Ex. 1001, 36:5–9. Similarly, claim 19 depends from claim 17 and adds that the computer executable code further comprises instructions to carry out the same steps. *Id.* at 38:13–18. Petitioner asserts that Chapman discloses these limitations. Pet. 41–43, 56–57 (citing Ex 1006, 16; Ex. 1016 ¶¶ 150, 151, 195, 196).

f. Claim 6

Claim 6 depends from claim 1 and further recites "receiving request to update database blocks associated with the VDB from the database server running on the system." Ex. 1001, 36:10–13. Petitioner contends that Sanders discloses this limitation. Pet. 43–44 (citing Ex 1004, 3, 5, 29; Ex. 1016 ¶¶ 153, 155).

g. Claim 8

Claim 8 depends from claim 1 and further recites provisioning a second virtual database system in a manner similar to the claim 1 limitation of

provisioning a virtual database. Ex. 1001, 36:25–32. Relying on the testimony of Dr. Zadok, Petitioner asserts that the combination of Sanders, Edwards, and Chapman meets the limitations of claim 8 for reasons similar to those asserted in connection with the claim 1 limitation of provisioning a virtual database. Pet. 44–46 (citing Ex. 1016 ¶¶ 156–161, 163–165).

h. Independent claims 14 and 24

Claim 14 differs from claim 1 only in that it claims a "method for backup of source databases using a virtual database system" that includes the step of performing backup of database blocks stored on a first storage system rather than the claim 1 limitation of "replicating the database blocks of the source database from the first storage system to a second storage system." Ex. 1001, 37:17–36.

Petitioner asserts that "Sanders in view of Chapman discloses replicating or transmitting snapshots comprising data blocks associated with a source database from a first SnapMirror destination to a second SnapMirror destination" and "replicating or transmitting snapshots associated with a source database from a first SnapMirror destination to a second SnapMirror destination to perform backup." Pet. 51 (citing Ex. 1004, 6; Ex. 1006, 6; Ex. 1016 ¶ 178). Accordingly, Petitioner contends that Sanders and Chapman disclose this limitation and the combination of Sanders, Edwards, and Chapman renders claim 14 obvious. *Id.* (citing Ex. 1016 ¶ 180).

Independent claim 24 is directed to a "computer program product having a non-transitory computer-readable storage medium storing computer-executable code for performing backup of source databases using a virtual database system," wherein the code comprises instructions to perform the steps recited by the method of claim 14. Ex. 1001, 39:9–40:7. Petitioner

argues that this claim format "does not materially distinguish Claim 24 from Claim 14 in any way with respect to invalidity analysis." Pet. 58. Thus, Petitioner asserts that, "for the reasons discussed above in connection with Claim 14, Sanders, in view of Edwards and Chapman, discloses and renders obvious the computer program product elements of Claim 24." *Id.* (citing Ex. 1016 ¶ 200).

i. Claims 16 and 25

Claim 16 depends from claim 14 and further recites linking a second source database, loading a plurality of point-in-time copies of the second source database, and performing backup of database blocks similarly to how such steps are carried out in claim 14 with respect to a first source database. Ex. 1001, 37:39–53. Similarly, claim 25 depends from claim 24 and adds that the computer executable code further comprises instructions to link a second source database, load a plurality of point-in-time copies of the second source database, and perform backup of database blocks. *Id.* at 40:8–25.

For the reasons relied on in connection with the parallel limitations of claims 14 and 24, Petitioner asserts that the combination of Sanders, Edwards, and Chapman renders claims 16 and 25 obvious. Pet. 51–54, 59 (citing Ex. 1016 ¶¶ 182–186, 203).

5. Patent Owner's Arguments

Patent Owner argues that the references relied on in the Petition describe physical disk storage systems rather than database systems. Prelim. Resp. 17–27. In particular, Patent Owner argues that WAFL is not a database system, but a file system. *Id.* at 20–22. Similarly, Patent Owner contends that NetApp's FlexVol allows physical storage on disks to be divided into logical volumes and thus creates storage volumes, not databases. *Id.* at 23–24.

According to Patent Owner, NetApp's SnapMirror feature is a file-system based mirroring method for disaster recovery of physical disk drive, which does not mirror databases. *Id.* at 24–26. Finally, Patent Owner contends that NetApp FlexClones are disk volumes, not databases. *Id.* at 26–27.

We are not persuaded by these arguments. The fact that the NetApp references describe technology that operates on file systems does not preclude operation on databases as well. For example, Sanders describes a system "to clone a DB2 database quickly and easily." Ex. 1004, 1. Sanders also describes FlexClone as a technology that delivers "a near-instantaneous point-in-time copy of [a] production database." *Id*. at 3. Furthermore, notwithstanding Patent Owner's arguments, the challenged claims do not preclude a database system operating directly on file system data blocks.

Patent Owner also argues that none of the references relied on in the Petition disclosed the claimed use of "database blocks" or the claimed "virtual database." Prelim. Resp. 27–50. Patent Owner bases its arguments on the alleged fundamental differences between the NetApp system and the claimed invention. *Id.* According to Patent Owner, the disk blocks disclosed in Edwards are not "database blocks" because they do not include metadata. *Id.* at 33–34. Furthermore, Patent Owner contends that the NetApp references "implement an operating system (Data ONTAP) and associated file system (WAFL) that are accessed in a traditional, disk-oriented, way" that does not provide a "virtual database." *Id.* at 38. Patent Owner further argues that the FlexVol technology simply creates a level of indirection in physical storage systems, but does not disclose a virtual database. *Id.* at 42–43. Patent Owner argues that the SnapMirror technology does not "virtualize" databases of files because mirroring produces an exact copy where "there is no decoupling of

the logical use of the files by a database and their physical implementation." *Id.* at 43–44. Finally, Patent Owner argues that a FlexClone is not a virtual database. *Id.* at 49–50.

All of these arguments rely on Patent Owner's overly narrow construction of the terms "database block" and "virtual database." For purposes of this Decision, we have construed those terms such that "database block" does not require metadata and "virtual database" does not require that the database be totally untethered from a physical storage system. Accordingly, these arguments are not persuasive.

Patent Owner also argues that even under Petitioner's proposed construction of "database blocks," Petitioner fails to show that the NetApp References properly disclose that a "database block" is "used by a database" as claimed. *Id.* at 37–38. Patent Owner bases this argument on the alleged distinction between a "database block" and a "disk block." *Id.* at 38 ("Disk blocks are a construct neither known by a database server nor used by a database server, but represent only an operating system construct for storing data on physical disks."). On this record, however, we are not persuaded that any alleged characteristics patentably distinguish an operating system disk block from a "database block." Thus, we are not persuaded that the claims preclude operating on file systems that contain databases as asserted by Patent Owner.

In addition, Patent Owner argues that one of ordinary skill in the art would not have been motivated to combine Sanders, Edwards, and Chapman because Sanders teaches against the proposed combination. *Id.* at 55. Patent Owner points to a passage in Sanders stating "[i]t is recommended not to use these automatically created Snapshot copies to create a clone volume" as

allegedly warning against using Snapshot copies to create clones as suggested by Petitioner. *Id.* (quoting Ex.1004, 23 (emphasis added by Patent Owner omitted)).

This argument is not persuasive because it does not account fully for the teachings of Sanders. For example, the cited passage of Sanders states "Snapshot copies are *also created automatically* for SnapMirror update purpose. It is recommended *not to use* these *automatically created Snapshot copies* to create a clone volume." Ex. 1004, 23 (emphases added). In a preceding passage, Sanders describes using other Snapshot copies (i.e., Snapshot copies not created for updating) to clone a database. *Id.* at 22 ("to create a clone database . . . you need to create Snapshot copies of the FlexVol volumes on the SnapMirror source storage system and update the SnapMirror relationship manually"). Hence, a reasonable reading of the sentence cited by Patent Owner would instruct users not to use the automatically created Snapshot copies, but, rather, to use the manually created copies when creating a clone volume. In fact, the very next paragraph after the paragraph cited by Patent Owner states "[a] clone volume can be created from the Snapshot copy by executing the following command on the destination storage system." *Id.* at 23.

Last, Patent Owner asserts that it

> disclosed to the Patent Office every NetApp feature that Petitioner now cites in the Petition. *See* Exs. 1009–1013. Petitioner attempts to take a second bite at the apple by submitting redundant references describing those same NetApp features. [Patent Owner] should not be required to retread the same ground after meeting its obligations to disclose these features during prosecution.

Prelim. Resp. 56. We are not persuaded that substantially the same references and arguments presented in this Petition were presented to the Office during prosecution because of the difference in the level of detail. Accordingly, on this record, the prosecution history does not present a sufficient reason to deny institution. *See* 35 U.S.C. § 325(d).

On balance, at this point in the proceeding, the evidence of record supports a conclusion that Petitioner has established a reasonable likelihood of prevailing on its assertion that claims 1–6, 8, 14, 16–19, 24, and 25 would have been obvious over the combination of Sanders, Edwards, and Chapman.

C. Conclusion

For the foregoing reasons, we determine that the information presented in the Petition establishes that there is a reasonable likelihood that Petitioner would prevail with respect to claims 1–6, 8, 14, 16–19, 24, and 25 of the '361 patent.

Any discussion of facts in this Decision are made only for the purposes of institution and are not dispositive of any issue related to any ground on which we institute review. The Board has not made a final determination on the patentability of any challenged claims. The Board's final determination will be based on the record as fully developed during trial.

IV. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that *inter partes* review is authorized on the following ground of unpatentability asserted in the Petition:

Claims 1–6, 8, 14, 16–19, 24, and 25 under 35 U.S.C. § 103(a) as unpatentable over Sanders, Edwards, and Chapman;

FURTHER ORDERED that pursuant to 35 U.S.C. § 314(a), inter partes review of the ’361 patent is hereby instituted commencing on the entry date of this Order, and pursuant to 35 U.S.C. § 314(c) and 37 C.F.R. § 42.4, notice is hereby given of the institution of a trial; and

FURTHER ORDERED that the trial is limited to the grounds identified above, and no other ground set forth in the Petition as to any challenged claim is authorized.

PETITIONER:

Robert Steinberg
Jonathan Link
Latham & Watkins LLP
bob.steinberg@lw.com
jonathan.link@lw.com

PATENT OWNER:

David Hadden
Saina Shamilov
FENWICK & WEST LLP
dhadden-ptab@fenwick.com
sshamilov-ptab@fenwick.com