

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

TESSELL, INC.,
Petitioner,

v.

NUTANIX, INC.,
Patent Owner.

Case No. IPR2025-00732
Patent No. 10,817,157

**PETITION FOR INTER PARTES REVIEW
UNDER 35 U.S.C. §§ 311-319 AND 37 C.F.R. § 42.1 et seq**

TABLE OF CONTENTS

MANDATORY NOTICES.....	xii
A. Real Party-In-Interest	xii
B. Related Matters.....	xii
1. United States Patent & Trademark Office	xii
2. United States Patent Trial and Appeal Board	xii
3. U.S. District Court for the Northern District of California.....	xii
C. Counsel and Service Information - § 42.8(b)(3) and (4).....	xiii
I. INTRODUCTION	1
II. STANDING CERTIFICATION.....	6
III. GROUNDS FOR UNPATENTABILITY	6
IV. THE '157 PATENT	6
A. Described Embodiments.....	6
B. Prosecution History	11
C. Person of Ordinary Skill in the Art (“POSA”).....	11
V. CLAIM INTERPRETATION	12
VI. <u>GROUND 1</u> : CLAIMS 1-19 ARE UNPATENTABLE OVER SIVASUBRAMANIAN+ZHA+SHEKAR	12
A. Sivasubramanian (EX1005).....	12
1. Overview	12
a. Database-Provisioning UIs	16
i. GUI	17
ii. Command-Line	21
b. Backup “Workflow”	23
2. Motivation to Customize Sivasubramanian’s UI	24
B. Zha (EX1006)	25
C. Shekar (EX1007)	27
D. Sivasubramanian+Zha+Shekar Combination.....	29
1. Sivasubramanian in view of Zha.....	29

a.	UI for retention period	35
b.	UI for protection schedule.....	37
2.	Sivasubramanian+Zha in view of Shekar	38
E.	Claim-by-Claim Analysis.....	42
1.	Claim 1	42
a.	[1Pre]: “A system comprising:”	42
b.	[1A]: “a memory storing computer-readable instructions in a virtual computing system; and”	43
c.	[1B]: “a processor executing the computer-readable instructions to:”	44
d.	[1C]: “present a dashboard to receive a user input in a graphical user [interface] (GUI) to provision a source database;”	45
e.	[1D]: “upon receiving the user input, present options in a first user interface of the GUI to receive a first selection of a database engine type;”	48
f.	[1E]: “upon receiving the first selection of the database engine type, present a plurality of selectable options to be selected via at least one user interface of the GUI, the plurality of selectable options comprising at least a plurality of:”	51
g.	[1F]: “a second selection of a first profile for creating a source database virtual machine;”	57
h.	[1G]: “a third selection of a second profile for creating the source database;”	59
i.	[1H]: “a fourth selection of a Service Level Agreement (“SLA”); and”	60
j.	[1I]: “a fifth selection of a frequency of capturing a snapshot and transactional log;”	61
k.	[1J]: “upon receiving the selections of the database engine type and the at least plurality of the first profile, the second profile, the SLA, and the frequency of capturing the snapshot and transactional log, create the source	

database and associate the created source database with the source database virtual machine; and”	61
1. [1K]: “display a status of the created source database on the dashboard.”	63
2. Claim 2	63
3. Claim 3	65
a. “software profile”	65
b. “compute profile”	67
c. “network profile”	68
4. Claim 4	69
5. Claim 5	70
6. Claim 6	72
7. Claim 7	76
8. Claim 8	78
9. Claim 9	78
10. Claim 10	79
a. [10PRE]: “A method comprising:”	79
b. [10A]-[10I]	79
11. Claims 11-14	80
12. Claim 15	81
a. [15Pre]: “An apparatus comprising:”	81
b. [15A]-[15K]	82
13. Claims 16-19	82
VII. <u>GROUND 2: CLAIMS 1-19 ARE UNPATENTABLE OVER</u> SIVASUBRAMANIAN+ZHA+SHEKAR+RDS	83
A. RDS (EX1013)	83
B. Sivasubramanian+Zha+Shekar+RDS Combination.....	83
C. Claim-by-Claim Analysis.....	85
1. Independent Claims 1, 10, and 15	85
2. Claims 4-5, and 12-13	86

3. Claims 2-3, 6-9, 11, 14, and 16-19.....	86
VIII.NO BASIS EXISTS FOR DISCRETIONARY DENIAL	87
A. 314(a).....	87
1. Stay Potential.....	87
2. Trial Timing	87
3. Litigation Investment	87
4. Issue overlap.....	88
5. Litigation Defendants	88
6. Other Circumstances	88
B. 325(d).....	88
1. Part One	88
2. Part Two	89
IX. CONCLUSION.....	90
X. CLAIM LISTING	91

TABLE OF AUTHORITIES

CASES

<i>Advanced Bionics, LLC v. Med-El Elektromedizinische Gerate GMBH,</i> IPR2019-01469, Paper 6 (PTAB Feb. 13, 2020)	88, 89, 90
<i>Apple Inc. v. Fintiv, Inc.,</i> IPR2020-00019, Paper 11 (PTAB Mar. 20, 2020).....	87
<i>Apple v. Seven Networks,</i> IPR2020-00156, Paper 10 (PTAB June 15, 2020)	87, 88
<i>Becton, Dickinson & Co. v. B. Braun Melsungen AG ,</i> IPR2017-01586, Paper 8 (PTAB Dec. 15, 2017).....	89, 90
<i>BMW of North America, LLC v. Michigan Motor Techs., LLC,</i> IPR2023-01224, Paper 15 (PTAB Feb. 15, 2024)	87
<i>Cellco P'ship v. Huawei Device Co.,</i> IPR2020-01117, Paper 10 (PTAB Feb. 3, 2021)	89
<i>Group III Int'l, Inc. v. Targus Group Int'l,</i> IPR2021-00371, Paper 21 (PTAB July 9, 2021).....	90
<i>Koninklijke Philips N.V. v. Google LLC,</i> 948 F.3d 1330 (Fed. Cir. 2020)	30
<i>Sand Revolution II v. Continental Intermodal Group-Trucking,</i> IPR2019-01393, Paper 24 (PTAB June 16, 2020).....	87, 88

STATUTES

35 U.S.C. § 102(a)	6
35 U.S.C. § 103	6
35 U.S.C. § 282(b)	11
35 U.S.C. § 314(a)	87
35 U.S.C. § 325(d)	89

REGULATIONS

37 C.F.R. § 42.100(b) 12

37 C.F.R. § 42.104(a).....6

OTHER AUTHORITIES

Patent Trial and Appeal Board Rules of Practice for Briefing Discretionary Denial
Issues, and Rules for 325(d) Considerations, Instituting Parallel and Serial
Petitions, and Termination Due to Settlement Agreement,
89 Fed. Reg. 28693 (Apr. 19, 2024)..... 89, 90

APPENDIX LISTING OF EXHIBITS

Exhibit	Description
1001	U.S. Patent No. 10,817,157
1002	Prosecution History of U.S. Patent No. 10,817,157
1003	Declaration of Dr. Samrat Bhattacharjee (“Bhattacharjee”)
1004	Curriculum Vitae of Dr. Bhattacharjee
1005	U.S. Patent No. 8,713,060 (“Sivasubramanian”)
1006	U.S. Patent No. 8,150,808 (“Zha”)
1007	U.S. Patent App. Pub. No. 2018/0232142 (“Shekar”)
1008	U.S. Patent No. 6,542,893 (“Quernemoen”)
1009	The American Heritage Dictionary of the English Language (5th ed. 2011) (“Heritage”)
1010	U.S. Patent No. 7,873,684 (“Souder”)
1011	U.S. Patent App. Pub. No. 2018/0324279 (“Barton”)
1012	Using Oracle Database Cloud Service – Creating a Database Deployment (2017) (“Oracle”) (Wayback Machine Capture, January 18, 2017), available at https://web.archive.org/web/20170118071307/https://docs.oracle.com/en/cloud/paas/database-dbaascloud/csdbi/create-db-deployment.html
1013	Amazon Relational Database Service User Guide (2016) (“RDS”) (Wayback Machine Capture, April 12, 2016), available at https://web.archive.org/web/20160412105355/http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/rds-ug.pdf
1014	U.S. Patent App. Pub. No. 2009/0319582 (“Simek”)
1015	U.S. Patent No. 7,331,000 (“Ohno”)
1016	U.S. Patent App. Pub. No. 2016/0182328 (“Bhasin”)
1017	U.S. Patent App. Pub. No. 2016/0077925 (“Tekade”)
1018	U.S. Patent App. Pub. No. 2019/0324865 (“Weissman”)
1019	RESERVED
1020	RESERVED

Exhibit	Description
1021	RESERVED
1022	“Taking a Backup Using Snapshots” (“LVM-Howto”) (Wayback Machine Capture, January 20, 2007), available at https://web.archive.org/web/20070120200004/https://tldp.org/HOWTO/LVM-HOWTO/snapshots_backup.html
1023	U.S. Patent App. Pub. No. 2004/0059878 (“Madany”)
1024	U.S. Patent App. Pub. No. 2006/0161879 (“Lubrecht”)
1025	PCT Pub. No. WO 2010/106578 (“Utsunomiya”)
1026	U.S. Patent No. 10,855,554 (“Freeman”)
1027	U.S. Patent No. 11,182,372 (“Jain”)
1028	U.S. Patent App. Pub. No. 2016/0057077 (“Gomatam”)
1029	U.S. Patent No. 11,036,696 (“Higginson”)
1030	U.S. Patent No. 9,305,068 (“Esposito”)
1031	U.S. Patent No. 7,873,610 (“Poulsen”)
1032	“Creating Instances,” Google Cloud (“Google”) (Wayback Machine Capture, September 18, 2018), available at https://www.archive.org/web/20180918103721/https://cloud.google.com/sql/docs/mysql/create-instance
1033	U.S. Patent App. Pub. No. 2003/0046353 (“Chung”)
1034	RESERVED
1035	RESERVED
1036	U.S. Patent App. Pub. No. 2018/0173452 (“Hu”)
1037	U.S. Patent No. 10,078,465 (“Wu”)
1038	U.S. Patent No. 9,830,082 (“Srinivasan”)
1039	U.S. Patent App. Pub. No. 2016/0266913 (“Achilopoulos”)
1040	U.S. Patent App. Pub. No. 2017/0262232 (“Zhao”)
1041	U.S. Patent App. Pub. No. 2019/0116237 (“Gibson”)
1042	U.S. Patent No. 10,206,092 (“Mellquist”)
1043	U.S. Patent No. 11,243,703 (“Watkins”)

Exhibit	Description
1044	U.S. Patent App. Pub. No. 2015/0207683 (“Adogla”)
1045	RESERVED
1046	RESERVED
1047	U.S. Patent App. Pub. No. 2006/0107006 (“Green”)
1048	U.S. Patent App. Pub. No. 2008/0059894 (“Cisler”)
1049	U.S. Patent App. Pub. No. 2013/0110779 (“Taylor”)
1050	U.S. Patent App. Pub. No. 2016/0306840 (“Deshmukh”)
1051	U.S. Patent App. Pub. No. 2018/0121521 (“Bivol”)
1052	U.S. Patent No. 10,496,302 (“Gaurav”)
1053	RESERVED
1054	RESERVED
1055	U.S. Patent No. 8,335,765 (“McAlister”)
1056	RESERVED
1057	RESERVED
1058	Affidavit of Nathaniel E. Frank-White
1059	RESERVED
1060	U.S. Patent No. 11,120,011 (“Priebe”)
1061	U.S. Patent No. 7,634,679 (“Quintanillo”)
1062	U.S. Patent App. Pub. No. 2009/0276771 (“Nickolov”)
1063	U.S. Patent App. Pub. No. 2016/0188421 (“Karinta”)
1064	RESERVED
1065	U.S. Patent App. Pub. No. 2016/0378361 (“Uriel”)
1066	RESERVED
1067	U.S. Patent App. Pub. No. 2017/0228227 (“Winterfeldt”)
1068	U.S. Patent App. Pub. No. 2014/0337844 (“Jin-844”)
1069	U.S. Patent No. 9,582,297 (“Jin-297”)
1070	U.S. Patent No. 10,572,292 (“Bhandari”)

Exhibit	Description
1071	U.S. Patent App. Pub. No. 2015/0188775 (“VanDerWalt”)
1072	U.S. Patent App. Pub. No. 2018/0063088 (“Hardy”)
1073	U.S. Patent No. 4,742,482 (“Inskeep”)
1074	U.S. Patent App. Pub. No. 2015/0207682 (“Nichele”)
1075	U.S. Patent App. Pub. No. 2013/0232497 (“Jalagam”)
1076	U.S. Patent App. Pub. No. 2010/0017801 (“Kundapur”)
1077	U.S. Patent App. Pub. No. 2006/0029096 (“Babbar”)
1078	UK Patent Application No. GB2451943 (“Wren-Hilton”)
1079	U.S. Patent App. Pub. No. 2013/0238331 (“Mikan”)
1080	U.S. Patent App. Pub. No. 2016/0239637 (“Miller”)
1081	U.S. Patent App. Pub. No. 2016/0066201 (“Kerpez”)
1082	U.S. Patent App. Pub. No. 2018/0060119 (“Zamir”)
1083	U.S. Patent App. Pub. No. 2015/0188789 (“Jayaprakash”)
1084	U.S. Patent App. Pub. No. 2015/0207703 (“Gallagher”)
1085	U.S. Patent App. Pub. No. 2006/0161462 (“Sharma”)
1086	RESERVED
1087	U.S. Patent App. Pub. No. 2002/0082858 (“Heddaya”)
1088	<i>Supported Platforms: MySQL Database, available at https://web.archive.org/web/20181106221826/https://www.mysql.com/support/supportedplatforms/database.html</i> (“MySQL”)
1089-1100	RESERVED
1101	A screenshot of the Expedia webpage displaying the “Preferred class” dropdown menu in its expanded state, Expedia website (Wayback Machine Capture, July 15, 2018), available at https://web.archive.org/web/20180715133951%20https://www.expedia.com/#expand (“Expedia”)
1102-1105	RESERVED
1106	U.S. Patent No. 9,389,962 (“Yueh”)

Exhibit	Description
1107	U.S. Patent App. Pub. No. 2018/0307728 (“Crupi”)
1108	U.S. Patent App. Pub. No. 2015/0095813 (“Welicki”)
1109	U.S. Patent App. Pub. No. 2006/0161865 (“Scott”)
1110- 1121	RESERVED
1122	Order Granting Defendant’s Motion to Compel Arbitration, <i>Nutanix v. Tessell</i> , Case No. 24-cv-01729-AMO, Dkt. No. 23 (Mar. 12, 2025)
1123	Federal Court Management Statistics–Profiles, U.S. District Courts–Combined Civil and Criminal (December 2024)

MANDATORY NOTICES

A. Real Party-In-Interest

Petitioner is the Real Party-in-Interest.

B. Related Matters

1. United States Patent & Trademark Office

The following U.S. patent applications claim the benefit of priority to U.S. Patent No. 10,817,157:

(i) U.S. Patent Application No. 17/006,595 (U.S. Patent No. 11,320,978), filed August 28, 2020; (ii) U.S. Patent Application No. 17/727,554 (U.S. Patent No. 11,907,517), filed April 22, 2022; (iii) U.S. Patent Application No. 18/224,008, filed July 19, 2023; (iv) U.S. Patent Application No. 18/489,838, filed October 18, 2023.

2. United States Patent Trial and Appeal Board

Tessell, Inc. v. Nutanix, Inc., Case Nos. IPR2025-00298, IPR2025-00322, and IPR2025-00733, challenge other patents that are also asserted in the district-court proceeding below.

3. U.S. District Court for the Northern District of California

Nutanix, Inc. v. Tessell, Inc., Case No. 3:24-cv-01729 (NDCA) (case stayed pending arbitration. *See* EX1122).

C. Counsel and Service Information - § 42.8(b)(3) and (4)

Lead Counsel	Anant K. Saraswat, Reg. No. 76,050
Backup Counsel	Elisabeth H. Hunt, Reg. No. 67,336
Service Information	<u>E-mail:</u> ASaraswat-PTAB@wolfgreenfield.com EHunt-PTAB@wolfgreenfield.com <u>Post and hand delivery:</u> Wolf, Greenfield & Sacks, P.C. 600 Atlantic Avenue Boston, MA 02210-2206 <u>Telephone:</u> 617-646-8000 <u>Facsimile:</u> 617-646-8646

A power of attorney is submitted with the Petition. Counsel for Petitioner consents to service of all documents via electronic mail.

I. INTRODUCTION

Petitioner requests *inter partes* review and cancellation of claims 1-23 of U.S. Patent No. 10,817,157 (“the ’157 patent,” EX1001).

The ’157 patent discloses a “database system” that receives an input via a “dashboard” for “provisioning” a “source database.” EX1001, Abstract, 1:23-28. A “source database” is a database from which “snapshots,” *i.e.* point-in-time records of the database’s state, are taken to create backups. EX1001, 4:46-59, 5:19-23; Bhattacharjee (EX1003), ¶¶28-29. “Provisioning” may involve either “creating” a database with user-specified parameters or “associating” an existing database with the database system. EX1001, 3:32-34; Bhattacharjee, ¶33.

The patent describes a “user interface [UI] that solicits information from and conveniently walks the user through the various steps for creating a new database.” EX1001, 4:8-11; Bhattacharjee, ¶49. The information solicited includes parameters such as the database software type (“engine”), database name, and a frequency for taking snapshots. EX1001, 14:14-25, 24:15-25:17, 5:64-67, Figs. 4C, 4E, 4F; Bhattacharjee, ¶41. The patent’s UI purportedly enables database creation “within minutes,” while database creation in prior-art systems allegedly required “multiple days” of manual work by “multiple entities.” EX1001, 3:32-55, 4:8-19. The patent also “provides the ability to register an existing database with

the database system” to let the user take advantage of “database management services offered by the database system.” EX1001, 4:29-44.

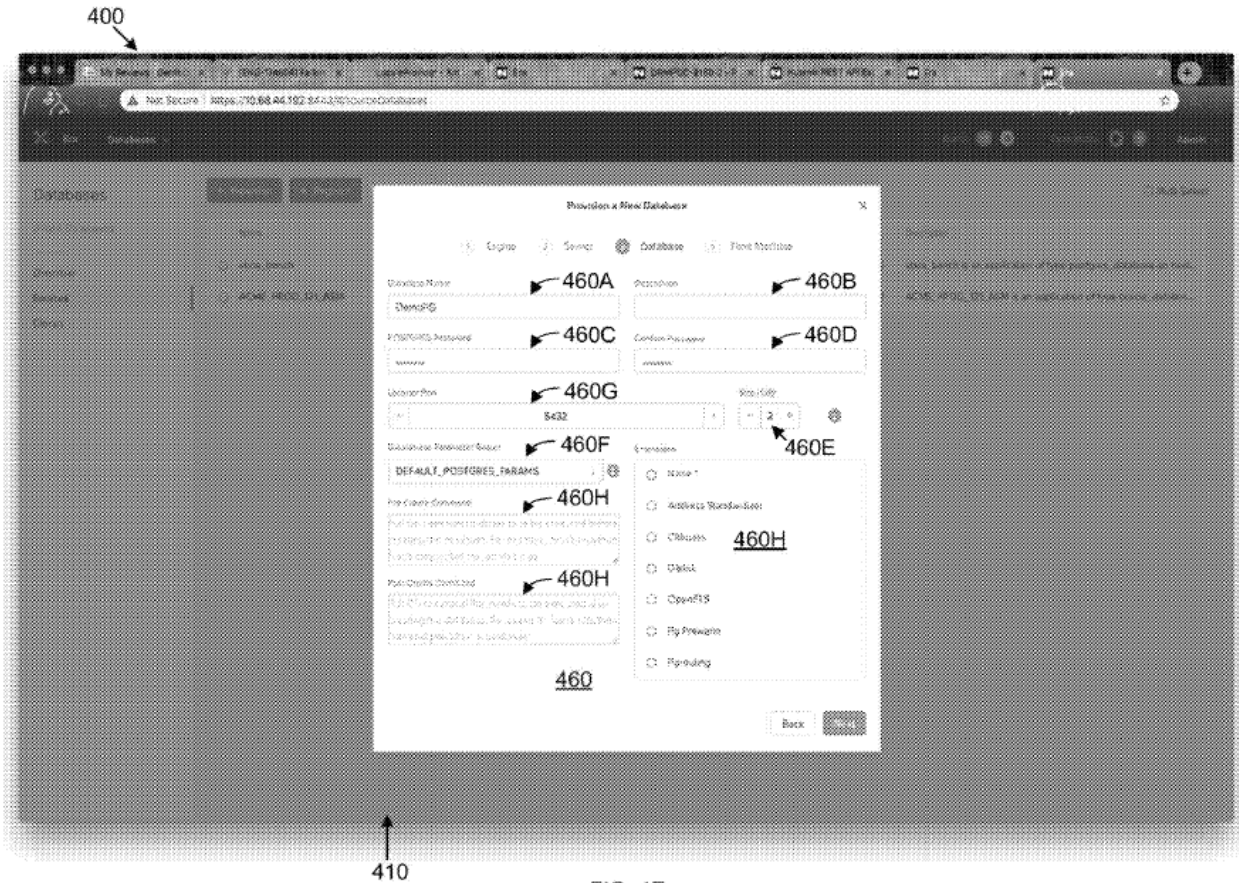


FIG. 4E

EX1001, Fig. 4E¹

Notably, the '157 patent does not purport to disclose any novel database implementation, *e.g.*, any new database software or any new data storage technology. Rather, the patent's claims recite simply accepting various parameters

¹ The blurry text is in the '157 patent's original figures.

via a UI and creating a database based on those parameters. Bhattacharjee, ¶¶54-55.

As this Petition demonstrates, all the input parameters recited in the claims were well-known database parameters, as disclosed in, *e.g.*, Sivasubramanian (EX1005) and Zha (EX1006). Nor was there anything novel about receiving the recited parameters as inputs and provisioning a database based on them. As discussed below, years before the claimed 2018 priority date, UIs providing for convenient and quicker parameter input and database provisioning were known.

For example, *in 2009*, Sivasubramanian, assigned to Amazon, disclosed a “control plane” that automates various “workflows” to replace “tedious manual procedures” previously required for tasks such as “provisioning and scaling data storage.” Sivasubramanian, Abstract, 1:46-56, 2:17-21. In contrast to the previously-used manual procedures, which “[took] a significant amount of time and effort,” a user could use a UI to “submit a call including information such as a type of hardware and a version of a database product” to Sivasubramanian’s control plane, which then provisioned the database; thus, Sivasubramanian provided the same functionality *9 years earlier* than the ’157 patent.

Sivasubramanian, 9:49-61, Fig. 5; Bhattacharjee, ¶¶50-52. Similar database-provisioning UIs were used in commercial products, *e.g.*, Amazon’s Relational

Database Service and Google's Cloud SQL. Amazon Relational Database Service User Guide ("RDS," EX1013), 13-18, 22; EX1032, 1-3; Bhattacharjee, ¶53.

Web Browser

FILE EDIT VIEW FAVORITES TOOLS HELP

Address GO

Control interface page

Repository name	Repository A			502
Task	504 ☒ Provision ☐ Create ☐ Delete ☐ Modify			
Database engine	MySQL	Version:	5	506
Storage capacity	150	Units	GB	508
Partition scheme				
Security scheme		Version:	0	
Access key		Version:	0	

Sivasubramanian, Fig. 5

Specify DB Details

Instance Specifications

DB Engine: mysql

License Model: general-public-license

DB Engine Version: 5.6.19a

Review the **Known Issues/Limitations** to learn about potential compatibility issues with specific database versions.

DB Instance Class: - Select One -

Multi-AZ Deployment: - Select One -

Storage Type: - Select One -

Allocated Storage*: 5 GB

Provisioning less than 100 GB of General Purpose (SSD) storage for high throughput workloads could result in higher latencies upon exhaustion of the initial General Purpose (SSD) IO credit balance. [Click here](#) for more details.

Settings

DB Instance Identifier*:

Master Username*:

Master Password*:

Confirm Password*:

* Required

Cancel Previous **Next Step**

RDS, 15

As this Petition demonstrates, Sivasubramanian's database-provisioning UI was customizable to accept any suitable user-desired parameters, such as by supplementing Sivasubramanian's disclosed parameters with parameters disclosed in Zha. Furthermore, there was nothing novel about taking snapshots from a database, as this was well-known. In addition, there was nothing novel about deploying the database on a virtual machine as the claims require, as this was disclosed in Shekar (EX1007).

All the '157 patent's claims would have been obvious as demonstrated below.

II. STANDING CERTIFICATION

Petitioner certifies that the '157 patent is available for *inter partes* review and that Petitioner is not barred or estopped from requesting *inter partes* review of the challenged claims. 37 C.F.R. §42.104(a).

III. GROUNDS FOR UNPATENTABILITY

The table below identifies the references, applicable claims, and basis for each ground of unpatentability. Sivasubramanian, Zha, and Shekar are prior art under post-AIA §§102(a)(1)-(2). Zha and a descendant of Sivasubramanian were of record during prosecution, but neither was substantively discussed or applied in any rejection, nor was Sivasubramanian+Zha+Shekar combination ever considered. *Infra* §VIII.B.

Ground Number and Reference(s)		Claims	Basis
1	Sivasubramanian (EX1005), Zha (EX1006), Shekar (EX1007)	1-19	§103
2	Sivasubramanian (EX1005), Zha (EX1006), Shekar (EX1007), RDS (EX1013)	1-19	§103

IV. THE '157 PATENT

A. Described Embodiments

The '157 patent discloses a “database system.” EX1001, Abstract. Databases (which long predated the patent) are collections of organized data. Bhattacharjee, ¶¶28-31; EX1008, 1:36-37; EX1009, 462. Databases can be created and managed by different types of database software, *e.g.* “Oracle” and “MySQL,”

which the patent refers to as “database engine types.” Bhattacharjee, ¶31; EX1001, 14:14-25; Sivasubramanian, 2:44-47.

The ’157 patent’s “database system” includes a “database provisioning system” that provides “database provisioning services.” EX1001, 10:44-57; Bhattacharjee, ¶32. The patent states that “[d]atabase provisioning services involve creating and/or associating databases with the database system for management and use”—*i.e.*, the patent’s “provisioning system” can either “create” a new database and provide resources to it, or provide resources to an already-existing database. EX1001, 3:32-34, 15:56-59; Bhattacharjee, ¶¶33-39; EX1010, 2:46-50; EX1011, [0004]; EX1008, Abstract; EX1012, 2-4; RDS, 100-102. Accordingly, the patent describes two different “example user interfaces”—for “creating” and “registering,” respectively, a “database.” EX1001, 22:23-25.

“[T]o initiate a database provisioning service,” the user navigates to a “database page” having a “create button” and a “register button.” EX1001, 23:43-24:3, Fig. 4B (annotated below). The exemplary database-creation interface is shown in Figs. 4C-4G. EX1001, 22:25-26; Bhattacharjee, ¶40.

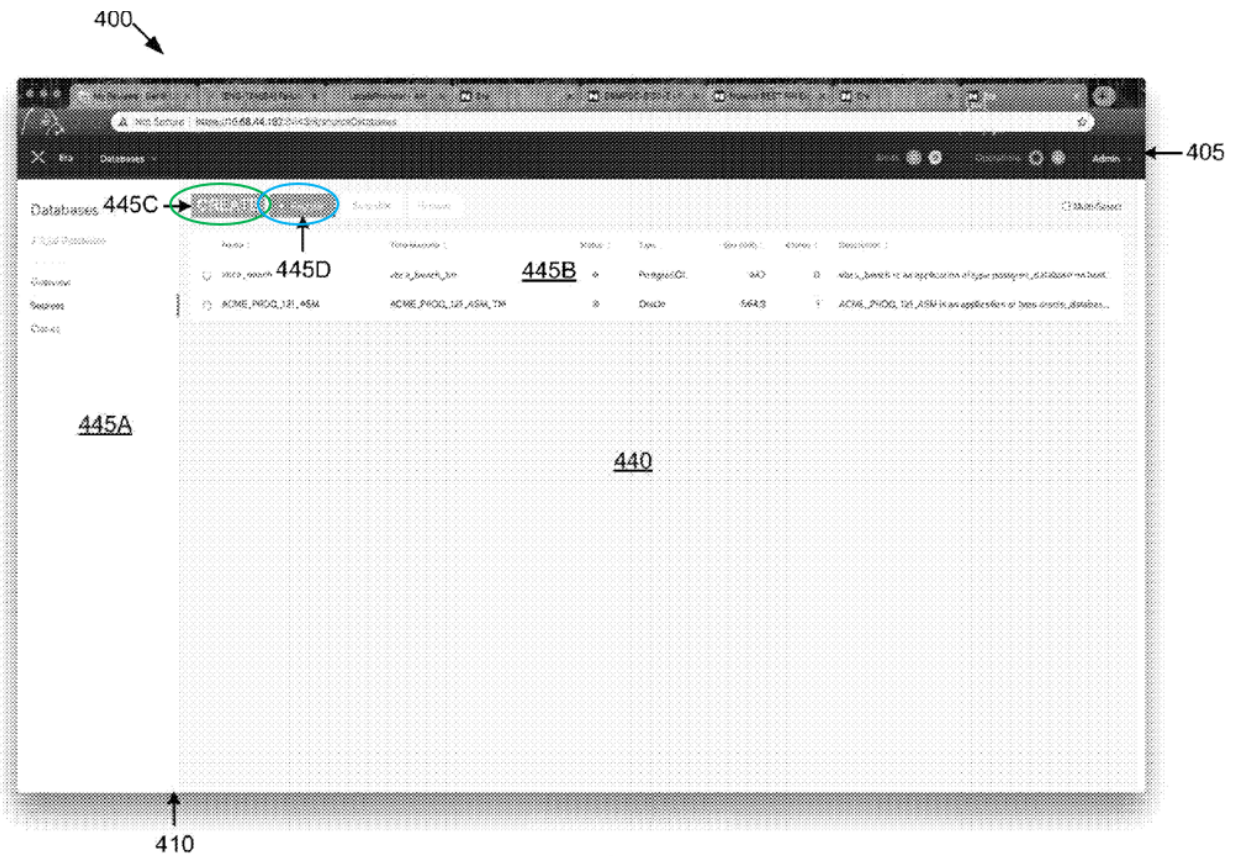
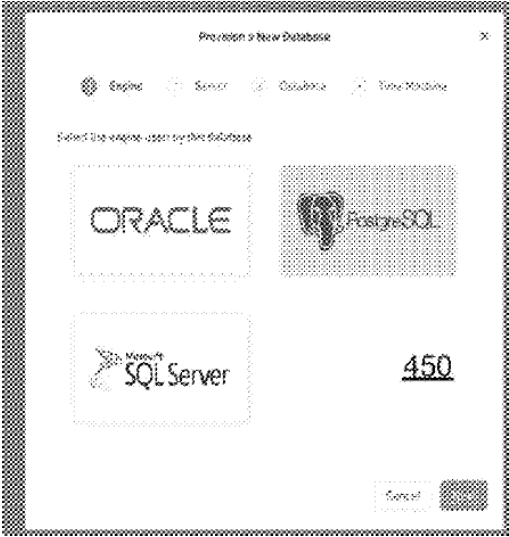
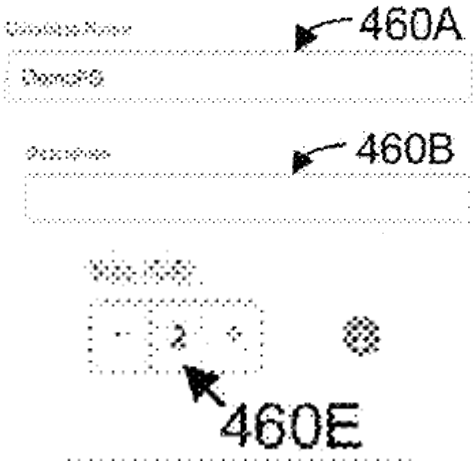
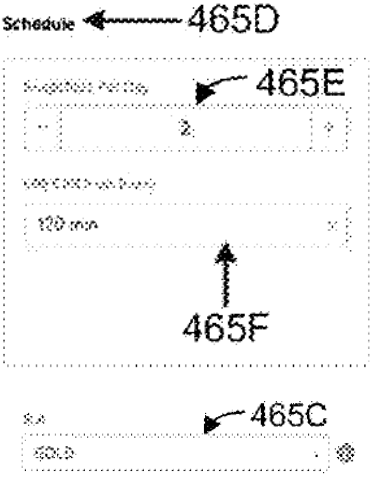


FIG. 4B

EX1001, Fig. 4B (annotated)

After pressing “create,” the user can enter various parameters on subsequent screens. EX1001, 24:4-10; Bhattacharjee, ¶41. Examples are shown below with excerpts from relevant figures.

'157 Specification	'157 Figures
“database engine type,” <i>e.g.</i>, “Oracle, PostgreSQL, etc.” (EX1001, 14:14-25, 24:15-23, FIG. 4C);	 FIG. 4C (excerpt)
“name 460A,” “description 460B,” “size 460E,” (EX1001, 24:49-63, Fig. 4E);	 FIG. 4E (excerpts)

'157 Specification	'157 Figures
“protection schedule 465D” and “SLA [service level agreement] level 465C” (EX1001, 25:1-17, Fig. 4F)	 FIG. 4F (excerpts)

Once the parameters are entered, the user presses a button to send the parameters to a “database creation system” to “start the database creation.” EX1001, 25:17-22. The database that is created can run on a virtual machine. EX1001, 3:21-25, 3:44-48, 6:12-18; Bhattacharjee, ¶42.

The user invokes UIs for “registering an existing database” by pressing “register” instead of “create” on the “database page.” EX1001, 25:43-54. The user can then enter various parameters on subsequent screens, including an “SLA and protection schedule.” EX1001, 25:60-67. “Upon receiving all of the parameters from the user, the database registration system... registers the database.” EX1001, 26:1-2; Bhattacharjee, ¶¶43-48.

B. Prosecution History

The '157 patent issued on October 27, 2020 from application number 16/228,728 filed on December 20, 2018, and claims no priority. EX1001, Face. During prosecution, the Examiner initially rejected all claims as obvious over Yueh (EX1106) in view of Crupi (EX1107). EX1002, 223. Following an amendment by the Applicant and an interview, the Examiner issued a Notice of Allowance accompanied by an Examiner's amendment that made extensive changes to the claim language, but did not explain any reasons for the changes and did not provide any reasons for allowance. EX1002, 279-286; Bhattacharjee, ¶¶56-57.

C. Person of Ordinary Skill in the Art ("POSA")

The '157 patent relates to UIs for deploying database services. EX1001, Abstract, 1:24-2:9, 3:44-49; *supra* §IV.A; Bhattacharjee, ¶¶58-59. A POSA at the time of the '157 patent's filing date (December 20, 2018) would have had a bachelor's degree in computer science, computer engineering, or a related field, with two years of academic and/or industry experience with database technologies. A higher level of education may substitute for less experience. Bhattacharjee, ¶¶60-62.

V. CLAIM INTERPRETATION

Claim terms are construed herein using the standard used in civil actions under 35 U.S.C. §282(b), in accordance with the ordinary and customary meaning as understood by a POSA and the prosecution history. 37 C.F.R. §42.100(b).

VI. GROUND 1: CLAIMS 1-19 ARE UNPATENTABLE OVER SIVASUBRAMANIAN+ZHA+SHEKAR

A. Sivasubramanian (EX1005)

1. Overview

Sivasubramanian concerns “the *creation, provisioning*, and management of data stores and instances” using a “control environment.” Sivasubramanian, Abstract, Fig. 3, 10:5-10; Bhattacharjee, ¶¶63-66. “[D]ata store’ refers to... any device or combination of devices capable of storing, accessing, and retrieving data”; such a device or devices may include “databases.” Sivasubramanian, 3:15-19.

Sivasubramanian’s “control environment” (a/k/a “control plane”) manages “workflows” that perform tasks in a “data plane” (a/k/a “data environment”); a “workflow” is a “sequence of tasks” for “specific job.” Sivasubramanian, Abstract, 2:21-24, 6:25-26, Fig. 2; Bhattacharjee, ¶¶67-68. “Workflows” may be initiated in response to requests submitted to the control plane using graphical or textual interfaces. Sivasubramanian, 2:26-32, 4:59, 5:4-10, 5:35-44, 6:12-23; *infra* §VI.A.1.a; Bhattacharjee, ¶¶69-73.

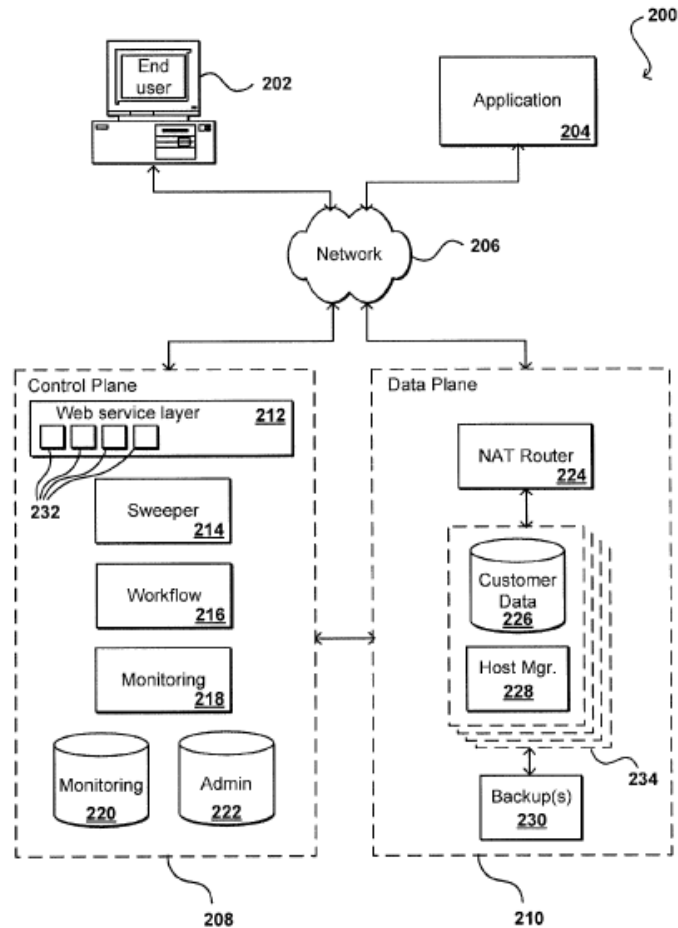


FIG. 2

Sivasubramanian, Fig. 2

One “workflow” the control plane can manage is “create database.”

Sivasubramanian, 6:59-65. As explained below, POSAs understood that Sivasubramanian’s “create database” workflow includes what the ’157 patent calls “provisioning” a database, for multiple reasons.

First, Sivasubramanian says the “create database” workflow “includes tasks such as *provisioning a data store instance*.” Sivasubramanian, 6:59-65.

Sivasubramanian uses “data instance” or “data store instance” to refer to a database together with the resources—such as processing power, memory, and storage—that the database uses. Sivasubramanian, 6:59-65, 11:48-51, 9:64-10:4; *see also* Sivasubramanian, 6:43-50 (workflow template for creating “**database**” based on user choice of “MySQL” v. “Oracle” “**instance**”); Bhattacharjee, ¶¶74-75.

Second, Sivasubramanian also references using a “Create**Database**” API (“application programming interface”) to “**provision**” a “**repository**,” and then using “Describe**Database**” to check if the “repository” is “provisioned.” 11:47-56, 2:26-32; Bhattacharjee, ¶76. POSAs understood that the “repository” is a *database* from the referenced API names “Create**Database**” and “Describe**Database**” and because Sivasubramanian describes a “database” as a type of “repository” and frequently uses those terms interchangeably. Sivasubramanian, 15:37-38 (“**repository** such as a relational *database*”), 4:58-62, 4:49, 5:15-20, 1:47-56, 11:52-56, 12:24-33, 13:18-21, 12:64-13:5, 13:27-33, Fig. 5; Bhattacharjee, ¶¶77-79. In Sivasubramanian, “workflows” may be initiated in response to “requests” submitted via “APIs”; thus, POSAs would have understood from the name that the “Create**Database**” API initiates the “create database” workflow. Sivasubramanian, 2:26-32, 5:4-10, 5:35-44, 6:12-23; Bhattacharjee, ¶80.

Third, the ’157 patent uses “provisioning” to encompass creating a database and giving it resources. *Supra* §IV.A. As noted above, Sivasubramanian discloses

the “create database” workflow “provisioning” a “repository,” *i.e.* database, and also provisioning a “data instance” representing a database and its resources.

POSAs understood from the name “create database” that the workflow “**creates**” a database and then gives it resources, *e.g.* the resources in the data instance; thus, Sivasubramanian’s “create database” workflow does the same thing the ’157 patent describes as “provisioning.” Bhattacharjee, ¶81.

Fourth, the ’157 patent says “[d]atabase provisioning services involve creating and/or associating databases with the database system **for management and use.**” EX1001, 3:32-34. Likewise, following completion of Sivasubramanian’s “create database” workflow that includes “provisioning,” the database, and the “data instance” on which it runs (as discussed above), can be **managed** by the control plane, and also can be **used**. *E.g.*, Sivasubramanian, 6:59-67 (““create database” workflow... might include[]” providing resources that “the customer can use to access... the data instance”), 9:4-8 (similar), 2:26-44 (similar), 9:45-65 (similar), 12:1-36 (describing API for modifying database), 13:18-15:31 (describing management actions available after creation), 19:32-64 (discussing workflow for backup); Bhattacharjee, ¶82.

Additionally, as noted *supra* §IV.A, the ’157 patent also uses the term “provisioning” to refer to “registering” an existing database and providing resources to it. EX1001, 3:32-34. The patent describes an exemplary “registering”

interface where a user identifies an existing database and then enters parameters for providing it resources. EX1001, 25:43-67, Fig. 5A-5E; Bhattacharjee, ¶83. As discussed *infra* §VI.A.1.a, Sivasubramanian likewise discloses UIs that let users identify an existing database and specify parameters for providing it resources.

a. Database-Provisioning UIs

Sivasubramanian says “F[igure] 5 illustrates an example of a display” that is an “example” of an “interface page... allow[ing] a user to *submit requests* to the *control plane*.” Sivasubramanian, 12:49-55. Sivasubramanian also describes Figure 5’s exemplary “display” and the below-referenced command-line tool as alternative ways to submit “calls to the control plane.” Sivasubramanian, 12:53-13:5. Thus, POSAs understood Sivasubramanian to disclose using a GUI like Figure 5’s to submit requests to invoke workflows. *Infra* §VI.A.1.a.i; Bhattacharjee, ¶84.

Sivasubramanian also discloses a “command line tool” that is an “example” way to “submit Web services calls to the control plane.” Sivasubramanian, 12:62-66. Sivasubramanian also says users can “call a Web service” to invoke the “‘CreateDatabase’ or similar API”; this API invokes the “create database” workflow (*see supra* §VI.A.1). Sivasubramanian, 11:47-52. Thus, POSAs understood that the command-line tool could be used to submit requests to invoke workflows. *Infra* §VI.A.1.a.ii; Bhattacharjee, ¶85.

As explained *infra* §§VI.A.1.a.i-VI.A.1.a.ii, the GUI and command-line, respectively, can be used to invoke workflows to “provision” a database (in the ’157 patent’s meaning) by creating a new database and allocating it resources and/or allocating resources to an existing database. Throughout this Petition, the term “database-provisioning UI” refers to a UI that performs either aforementioned “provisioning” process. Bhattacharjee, ¶86.

i. GUI

Sivasubramanian’s Figure 5 “display” lets the user “select an action to be performed 504,” and select “options for the actions” including “database engine” (506) and “version information” (508). Sivasubramanian, 12:53-62; Bhattacharjee, ¶87.

Web Browser

FILE EDIT VIEW FAVORITES TOOLS HELP

Address

GO

Control interface page

Repository name: Repository A

Task: ☒ Provision ☐ Create ☐ Delete ☐ Modify

Database engine: MySQL Version: 5

Storage capacity: 150 Units: GB

Partition scheme:

Security scheme: Version: 0

Access key: Version: 0

Save Cancel

Sivasubramanian, Fig. 5

In Figure 5 (annotated above), the “action to be performed 504” (Sivasubramanian, 12:59-60), labeled “**Task**,” shows the “**Provision**” option selected. POSAs understood that the “Provision” “action”/“Task” provisions a *database*, and that the “Repository name” is a database name, because the interface includes an option for selecting a “**Database engine**,” and because of Sivasubramanian’s repeated use of “repository” for “database” (noted *supra* §VI.A.1). Sivasubramanian, 12:56-62, Fig. 5; Bhattacharjee, ¶88. Furthermore, the figure shows that the “Repository [*i.e.*, database] name” field lets the user select from options via a drop-down menu (indicated by the down arrow),

indicating that the “Provision” “action”/“Task” provisions an already-*existing* database the user selects, rather than creating a new database. Bhattacharjee, ¶89; Sivasubramanian, 12:38-40.

As shown above, Figure 5 also includes a “Create” option. POSAs understood that the “Create” option presents an interface for creating a *database*, at least because (1) Sivasubramanian describes the GUI and the “rds-create-*database*” command-line command as alternative methods for creating a database, and (2) Sivasubramanian describes “*database* engine” as an “option” for the “action 504,” labeled “Task 504” in Figure 5 (annotated above). Sivasubramanian, 12:49-13:5; Bhattacharjee, ¶90.

Sivasubramanian says Figure 5’s interface “include[s] options that allow the user to enter information needed to perform a control action on the data plane.” Sivasubramanian, 12:55-57; Bhattacharjee, ¶91. Thus, POSAs would have understood Sivasubramanian to teach, or at minimum render obvious, that the fields for database-creation parameters presented in the “Create” interface would include fields for the parameters shown in Figure 5 for the “Provision” interface, as explained below.

Sivasubramanian’s create-database workflow includes provisioning the database by allocating it resources. *Supra* §VI.A.1. Thus, POSAs understood that the “Create” interface likewise included the fields on the “Provision” interface.

Bhattacharjee, ¶92. Further, Sivasubramanian discloses an “rds-create-database” command-line command for creating a database. Sivasubramanian, 13:1-5; *infra* §VI.A.1.a.ii. The parameters Sivasubramanian shows “rds-create-database” accepting include parameters shown in Figure 5’s “Provision” task screen, including name, engine, and size. Sivasubramanian, 13:1-5 (“rds-create-database” command accepting “dbname,” “size,” “engine”), Fig. 5 (showing fields for “Repository name,” Database engine,” and “Storage capacity”); Bhattacharjee, ¶93 (explaining POSAs understood “db” in “dbname” to be a common abbreviation for “database,” and understood “capacity” in GUI to be equivalent to “size” in “rds-create-database” command) (citing Sivasubramanian, 12:42-44).

Additionally, POSAs understood, or at minimum would have found obvious, that in the “Create” interface, the user could enter a new “Repository name,” *i.e.* database name as discussed above, for the new database being created, rather than selecting the name of a previously-created database to provision as in the “Provision” interface. Bhattacharjee, ¶94.

To the extent not taught by Sivasubramanian, POSAs would have found implementing the “Create” task GUI to accept inputs of parameters shown on the “Provision” task GUI to be obvious as a well-known UI implementation that beneficially increased user convenience by letting the “Create” UI accept parameters needed to provision a created database (functionality Sivasubramanian

itself teaches, as discussed above). Bhattacharjee, ¶95; RDS (EX1013), 13-18 (describing “Creating a MySQL DB Instance” UI that creates a “provisioned” database). Additionally, POSAs would have understood that accepting a new database name was logical in a UI for creating a new database. Bhattacharjee, ¶95. POSAs would have reasonably expected success implementing the “Create” UI in this manner because it was within POSAs’ skill. *Infra* §VI.A.2.

As discussed *supra* §VI.A.1.a, Sivasubramanian’s Figure 5 GUI can be used to submit requests to the control plane to initiate workflows. Thus, POSAs understood, or at least would have found obvious, that the “Create” option in the Figure 5 GUI invokes the “create database” workflow, which *provisions* a new database (under the ’157 patent’s meaning) as discussed *supra* §VI.A.1. POSAs also understood, or at least would have found obvious, that the “Provision” task likewise invokes a workflow for provisioning an existing database. Sivasubramanian, 5:16-20, 7:24-28; Bhattacharjee, ¶96.

ii. Command-Line

The command-line tool presents a text-based interface (command line) for the user to enter text in a specified command format to input parameters for a task such as database creation. Sivasubramanian, 12:64-13:5. The command-line format accepts input parameters as a list of pairs of parameter names and values, as shown in the example “rds-create-database” command:

```
rds-create-database --identifier customerprod --dbname customer --  
size 80 [-]-class small --engine mysql5.1 --master master_username --  
password master_password -port 4030
```

Sivasubramanian, 12:64-13:5; Bhattacharjee, ¶97.

Sivasubramanian teaches that the “rds-create-*database*” command creates a database, as indicated by the command’s name and by Sivasubramanian’s description of it as an example command for “generat[ing] a request to create a new data *repository*,” a term which in Sivasubramanian encompasses a “database,” as discussed *supra* §VI.A.1. Sivasubramanian, 12:64-13:5; Bhattacharjee, ¶98.

The command-line tool can be used to submit requests to the control plane to initiate workflows. *Supra* §VI.A.1.a. Thus, Sivasubramanian teaches, or at least renders obvious, that the “rds-*create-database*” command invokes the “create database” workflow, which *provisions* a new database (under the ’157 patent’s meaning) as discussed *supra* §VI.A.1. Bhattacharjee, ¶99.

Furthermore, POSAs would have understood the “rds-create-database” command as just one representative example discussed for illustration purposes, and would have been motivated to similarly implement a command-line version of the above-discussed “Provision” task in Figure 5’s GUI that provisions an existing database, as this would beneficially provide parallel functionality with the GUI. POSAs would have expected success because implementing such a command,

which may have been called, *e.g.*, “rds-provision-database,” was within a POSA’s skill as discussed *infra* §VI.A.2. Bhattacharjee, ¶100.

b. Backup “Workflow”

Sivasubramanian discloses a “recovery and backup” “workflow[.]”

Sivasubramanian, 6:31-42; Bhattacharjee, ¶101. Sivasubramanian says the control plane can use “software frameworks” for “database backup and recovery,” and also says a “framework” can handle “backing up both the data files and any associated log files.” Sivasubramanian, 16:21-27, 19:25-28. Sivasubramanian describes one exemplary “approach for backing up data” whereby a “su[s]pendDatabaseForBackup” command is used to “suspend[] data operations” while “s[n]apshots are taken of the appropriate data volumes.” Sivasubramanian, 19:32-41; Bhattacharjee, ¶102-103.

Additionally, in Sivasubramanian’s above-mentioned “approach for backing up data,” “log files” are “copied” while the database is suspended.

Sivasubramanian, 19:32-41. The log files are used for “replay[ing]... *transactions* in case the data files have to be *restored*”; thus, Sivasubramanian’s “log files” are what other art and the ’157 patent refer to as “transaction logs” or “transactional logs.” Sivasubramanian, 19:65-20:1; Zha, 10:6-11; RDS, 112; EX1001, 36:49-59; EX1060, 1:14-22, 1:61-63; EX1061, 4:17-30; Bhattacharjee, ¶104.

2. Motivation to Customize Sivasubramanian's UI

Although Sivasubramanian's Figure 5 shows a particular example set of configuration parameters that can be input, POSAs understood that the UI shown could beneficially accept additional inputs not expressly shown. Bhattacharjee, ¶105. Sivasubramanian says Figure 5 is an “*example* of a display” in an “interface” including “options that allow the user to enter *information needed* to perform a control action on the data plane,” and notes that “*other* such values” aside from those shown can be specified. Sivasubramanian, 12:53-64.

Sivasubramanian also teaches that the user interface to the control plane is flexible and enables the user to enter whatever options are appropriate to accomplish a desired workflow. Sivasubramanian, 11:49-52 (“The user can call a Web service to specify *any* desired values for an instance type..., storage size, repository name, port, *and other such values.*”)², 2:23-26, 11:41-47, 6:7-12, 6:45-52; Bhattacharjee, ¶¶106-107. Furthermore, Sivasubramanian's discussion of the “rds-create-database” command shows different options for parameters than those shown in Figure 5, indicating that other parameters could similarly be added to Figure 5's GUI. Sivasubramanian, 13:1-5 (“rds-create-database” command including parameters for “identifier,” “class,” “password”); Bhattacharjee, ¶108.

² Emphases added unless otherwise indicated.

Such UI implementations were well within a POSA's skill.

Sivasubramanian describes Figure 5 as an example page rendered in a "browser application." Sivasubramanian, 12:52-55. Sivasubramanian also discloses implementing its functionality on "[w]eb server[s]" running "applications" implemented in "any programming language." Sivasubramanian, 23:4-16. POSAs had the skill to use "programming language[s]" to implement an application to display a version of Figure 5's exemplary UI including additional options. Bhattacharjee, ¶¶109-110.

B. Zha (EX1006)

Zha teaches techniques for creating a "virtual database"—*i.e.*, a copy of an existing database from a particular point in time—used for purposes including "backup." Zha, Abstract; 6:49-59, 24:24-27; Bhattacharjee, ¶¶111-112. Zha's "[m]ethods of creating a virtual database (VDB) read different point-in-time copies of a source database," which Zha also refers to as a "production database." Zha, 1:66-2:6. Zha also teaches retrieving "transaction logs" from the source/production database; these logs can be combined with point-in-time copies to "reconstruct a copy of a database." Zha, 9:63-10:11, 15:58-62, 18:1-3; Bhattacharjee, ¶¶113-114.

Zha's "database storage system" (where database copies are stored) can be configured by a "database administrator" using a "system configuration manager."

Zha, 6:11-58, 11:26-28; Bhattacharjee, ¶¶115-116. This “manager” lets a user “setup... the *schedule*” specifying the “frequency and times during the day” for “retriev[ing] *PIT [point-in-time] copies of databases*” and “retriev[ing] updates to online *transaction logs*.” Zha, 11:33-44; Bhattacharjee, ¶116.

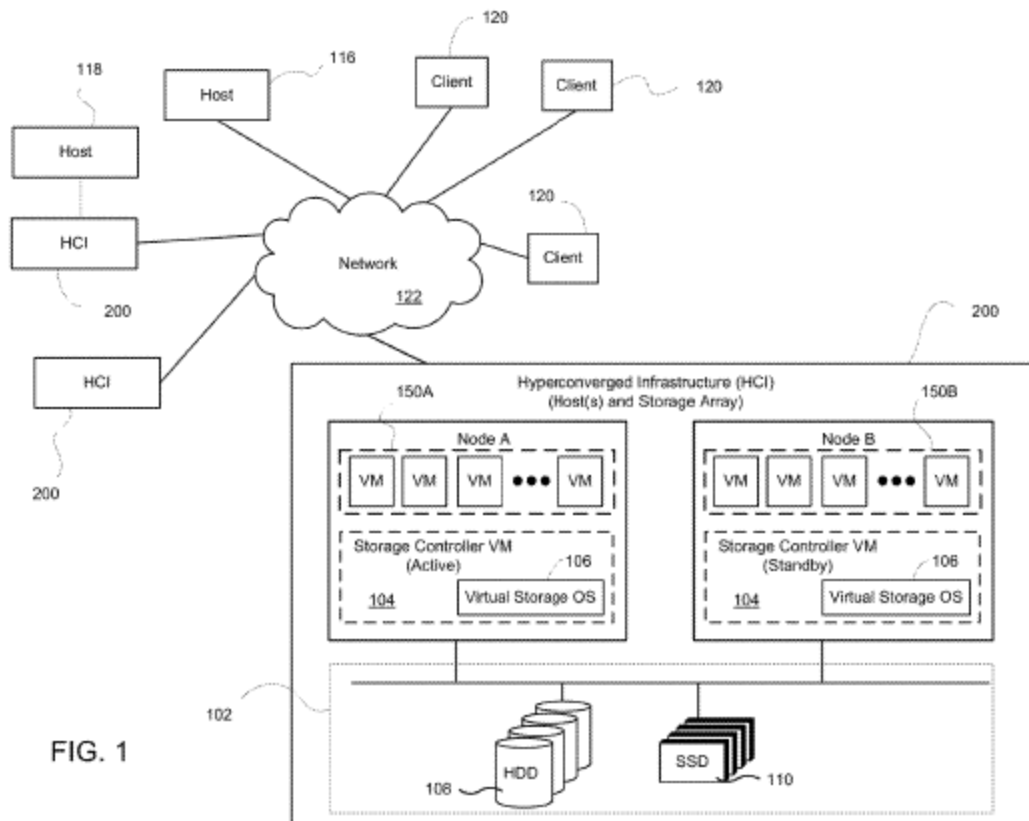
POSAs understood that Zha’s “point-in-time copies” are “snapshots” of the source database at the time the copy was made, for multiple reasons. The term “snapshot” generally referred to a record of the database’s state at a point in time. Bhattacharjee, ¶117; EX1014, [0002]; EX1015, 1:54-61. Furthermore, Zha states that “transaction logs can be applied to a VDB [virtual database] created based on a PIT copy... to create a *snapshot* of the source database for a time point that occurs *after* the PIT copy was made.” Zha, 17:63-66. Because applying transaction logs creates a “snapshot” of the database at a time *after* the PIT copy was made, POSAs understood that the PIT copy itself represents a “snapshot” *at* the time it was made, since Zha also explains that the transaction logs “can be used to reconstruct a *copy* of a database... corresponding to times in the past in *between* the times as which *point-in-time copies* are made.” Zha, 10:6-11; Bhattacharjee, ¶118.

Zha also discloses “policies defined by a database administrator” for operations including “purging of information... including point-in-time copies of databases and transaction log information.” Zha, 11:45-54. Such “policies”

include a “schedule for purging the point-in-time copy after a retention period” (Zha, claim 11, 35:18-20) and “a schedule for purging the transaction logs after a retention period” (Zha, claim 13, 35:24-26). Bhattacharjee, ¶119.

C. Shekar (EX1007)

Shekar discloses a “[h]yperconverged infrastructure” (“HCI”) providing “storage” and “compute” capabilities. Shekar, Title, [0014], [0032], Fig. 1. Shekar’s goal is to provide a “storage device” with “high performance.” Shekar, [0005]. Bhattacharjee, ¶120.



Shekar, Fig. 1

Shekar's HCI includes two "nodes" that include a "virtual storage connect module" connecting the node to a "storage array"; the array "provide[s] storage services to one or more host applications executing on the corresponding HCI 200." Shekar, [0035], [0037], [0043], Fig. 2A; Bhattacharjee, ¶¶121-122. The applications run on "guest virtual machines [VMs]" executing on each node's hardware. Shekar, [0008], [0038], Fig. 2A; Bhattacharjee, ¶123. Each node also includes a "storage VM" (or "storage controller VM"). Shekar, [0033], Fig. 2A. The storage VMs on the two nodes "cooperatively manage access to the storage array." Shekar, [0033], [0040]-[0042]; Bhattacharjee, ¶124.

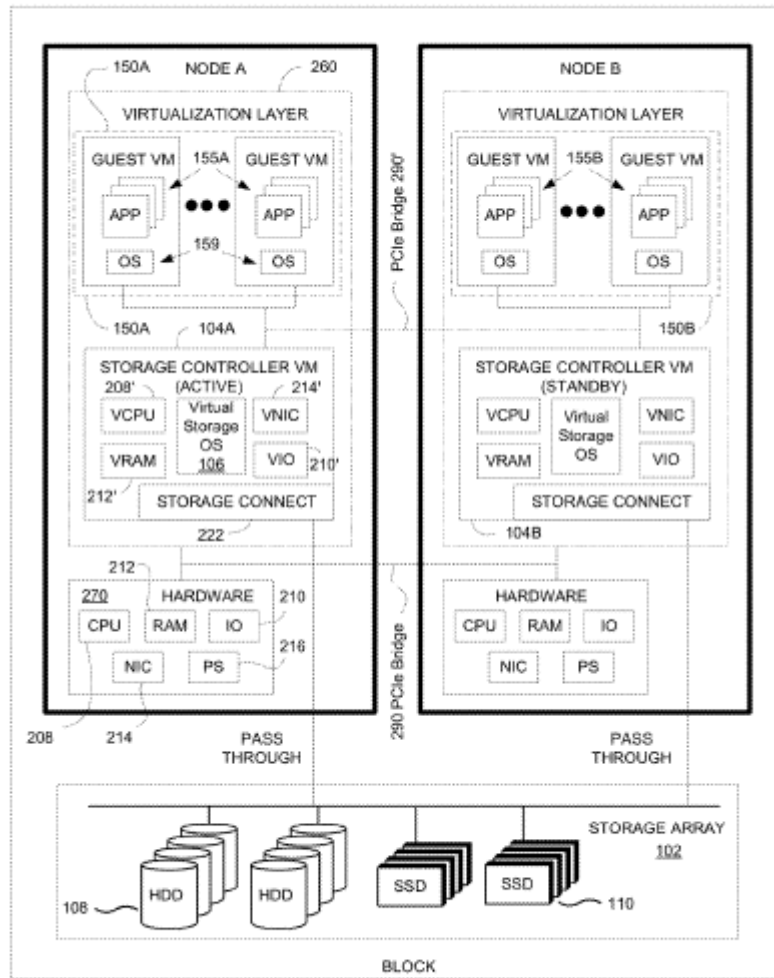


FIG. 2A

Shekar, Fig. 2A

D. Sivasubramanian+Zha+Shekar Combination

1. Sivasubramanian in view of Zha

POSAs seeking to implement Sivasubramanian's "backup" workflow discussed *supra* §VI.A.1.b would have considered references like Zha regarding certain implementation details Sivasubramanian is silent/agnostic about. Although Sivasubramanian and Zha use slightly different terminology, both discuss database

backup using snapshots and transaction logs, with Zha’s teachings supplying further details; thus, POSAs would have found it obvious to implement Sivasubramanian using Zha’s teachings in multiple independent ways, as discussed below. Bhattacharjee, ¶125. As also discussed below, POSAs would further have found it obvious to apply well-known aspects of POSAs’ background knowledge when implementing the combination. *See Koninklijke Philips N.V. v. Google LLC*, 948 F.3d 1330, 1338 (Fed. Cir. 2020) (affirming obviousness over named prior-art reference in view of POSAs’ background knowledge as evidenced by corroborating reference).

Sivasubramanian’s “snapshots” discussion appears in a paragraph describing “[an] approach for backing up data” that immediately follows a paragraph introducing Sivasubramanian’s discussion of “*repository* and data backup” and also noting that “*approaches* can differ from various *database engines*.” Sivasubramanian, 19:20-35. Sivasubramanian’s language and structure thus confirm that Sivasubramanian discloses *database* snapshots. Bhattacharjee, ¶126; *see also* Sivasubramanian, 15:1-15 (earlier discussion of “tak[ing] a snapshot of a

production instance” so that a “test instance” created with a “create-*database*” command can be “fully populated and comparable with the production [data]”).³

Additionally, Sivasubramanian’s exemplary “approach” uses a “su[s]pend*DatabaseForBackup*” command to “suspend[] data operations” while “*s[n]apshots* are taken of the appropriate data volumes.” Sivasubramanian, 19:32-41. Sivasubramanian also discloses storing a database’s data on “volume[s]” allocated to a database’s storage “instance.” Sivasubramanian, 6:59-63, 8:22-25, 15:56-60; Bhattacharjee, ¶127. POSAs understood that taking snapshots of underlying data volumes was a typical and customary way to take a database snapshot; thus, POSAs understood Sivasubramanian to disclose taking *database* snapshots by taking snapshots of the underlying data volumes. Bhattacharjee, ¶¶128-129 (citing corroborating EX1022, 1; EX1015, 1:28-29, 1:54-2:29; EX1055, 17:23-40)

POSAs understood that “tak[ing]” “snapshots” of a database in Sivasubramanian (Sivasubramanian, 15:1-5, 19:20-64) is analogous to what Zha refers to as “retriev[ing] PIT [point-in-time] copies” (Zha, 11:33-44, 17:63-66), and likewise understood that “backing up” the “log files” in Sivasubramanian

³ Sivasubramanian 15:3 says “production date,” but POSAs understood this is a typo. Bhattacharjee, ¶126.

(Sivasubramanian, 19:65-20:11) is analogous to what Zha refers to as “retriev[ing] updates to online transaction logs” (Zha, 11:33-44); both references’ processes result in the snapshot/log being stored in some storage location used for storing snapshots/logs. *Compare* Sivasubramanian, 19:32-35 with Zha, 2:19-24, 9:67-10:3; Bhattacharjee, ¶130.

Sivasubramanian does not restrict implementation details such as daily frequency of retrieving snapshots and transaction logs, what time of day these events occur, or retention duration of snapshots and logs. Sivasubramanian, 19:20-20:11 However, POSAs understood it was typical to collect snapshots, logs, and other database-backup information at user-configurable frequencies and times and retain such information for user-configurable durations. Bhattacharjee, ¶131 (citing corroborating EX1016, [0009]; RDS, 16, 111-12; EX1017, [0046], Fig. 6; EX1014, [0034], Fig. 2). Furthermore, POSAs understood that taking and storing snapshots and transaction logs required computing resources and storage space and potentially impacted the database’s performance; thus, POSAs understood that allowing users of Sivasubramanian’s control plane to specify snapshot and log frequencies, times, and retention periods beneficially enabled users to tune those parameters to suit their needs. Bhattacharjee, ¶132 (citing corroborating EX1023, [0023]; RDS, 5; Sivasubramanian, 19:31-41; EX1015, 1:54-61).

Therefore, POSAs seeking to implement Sivasubramanian’s exemplary “approach for backing up data” (Sivasubramanian, 19:32) would have been motivated to use an administrator-defined “schedule” including a “frequency” and “times during the day” for taking snapshots and backing up transactional/transaction logs, and to store them for administrator-defined “retention periods,” as Zha discloses. Zha, 11:33-54, 35:18-20, 35:24-26; Sivasubramanian, 19:20-20:11; Bhattacharjee, ¶133.

In the resulting “Sivasubramanian+Zha” combination, Sivasubramanian’s database-provisioning UIs discussed *supra* §VI.A.1.a accept user-defined snapshot and transaction-log collection frequencies, times during the day, and/or retention periods, as detailed *infra* §§VI.D.1.a-VI.D.1.b. Additionally, in the combination, the provisioning process would setup the database to have snapshots and logs (1) collected at the respective user-defined frequencies and times and (2) stored for the respective user-defined periods. Sivasubramanian, 19:32-39; Bhattacharjee, ¶134.

Alternatively, POSAs would have found it obvious to employ Zha’s technique of creating VDBs (alternatively or additionally to creating Sivasubramanian’s snapshots) to beneficially achieve the “efficient[] use [of] storage and computing resources” Zha’s technique provides. Zha, 1:63-65; Bhattacharjee, ¶135. POSAs would have been motivated to customize Sivasubramanian’s database-provisioning UIs to let the user set snapshot and log

schedules and retention periods for the same above-discussed reasons, and the resulting system would incorporate the UI elements detailed *infra* §§VI.D.1.a-VI.D.1.b into Sivasubramanian’s database-provisioning UIs. The resulting system would also setup the provisioned database to have “point-in-time copies,” *i.e.* snapshots, and “transaction logs” “retriev[ed]” from it, as Zha teaches, based on user-entered parameters as discussed above. Zha, 11:33-44; *supra* §VI.B; Bhattacharjee, ¶135.

POSAs would reasonably have expected success with either alternative above-discussed implementation of Sivasubramanian+Zha. As detailed *infra* §§VI.D.1.a-VI.D.1.b, POSAs had the skill to implement Sivasubramanian’s database-provisioning UIs to accept the inputs Zha teaches. POSAs also had the skill to have the provisioning process setup the database to have snapshots and logs taken/retrieved and retained based on user-entered parameters, since Sivasubramanian itself discloses the control plane executing tasks based on user-entered parameters. *E.g.*, Sivasubramanian, 11:41-52, 12:64-13:5, 19:32-39; Bhattacharjee, ¶¶136-138. Moreover, Sivasubramanian says the “approach” it describes for backup is “[a]*n*” approach, meaning that Sivasubramanian’s control plane could accommodate other approaches like Zha’s. Sivasubramanian, 19:32; Bhattacharjee, ¶139. POSAs had the skill to implement Sivasubramanian’s control plane to use Zha’s “virtual database” technique; Zha’s teachings can be

implemented in software, and Sivasubramanian likewise discloses implementing the control plane using software “framework[s]” that can perform tasks including “backup and recovery.” Bhattacharjee, ¶140; Zha, 32:49-52; Sivasubramanian, 16:21-28.

a. UI for retention period

It was well-known to specify a “service level” setting retention periods for database-protection-related data such as snapshots and transaction logs; such levels were often specified in a “service level agreement” (“SLA”). Bhattacharjee, ¶141; EX1017, [0046], Fig. 6 (“SLA protecting a database” including “snapshot policy” specifying retention period); EX1024, [0120]; EX1025, [0004]; EX1026, 11:39-43; EX1027, 19:32-35. It was also well-known to define multiple SLA “levels” named, *e.g.*, “gold,” “silver,” etc. Bhattacharjee, ¶142; EX1028, [0046], [0056]-[0058]; EX1029, 9:4-7; EX1030, 3:62-4:6.

In Sivasubramanian+Zha, POSAs would have been motivated to have control-plane users specify Zha’s configurable snapshot/log retention periods (*supra* §VI.B) by specifying an SLA level, for multiple reasons. POSAs understood that it was beneficial to give users familiar UIs, and that users were familiar with SLAs to specify retention periods, as discussed above (and also familiar generally with specifying time intervals). Bhattacharjee, ¶143; EX1017, Fig. 6; RDS, 15-16; EX1014, Fig. 2. Furthermore, POSAs understood that letting

users choose from predefined sets of parameter values was well-known and increased user convenience. Bhattacharjee, ¶144; EX1029, 9:49-10:5; EX1031, 1:16-29. Thus, in either above-discussed Sivasubramanian+Zha implementation, POSAs would be motivated to let the user specify transaction log and snapshot retention periods in the database-provisioning UIs discussed *supra* §VI.A.1.a via the well-known technique of SLA level selection, to beneficially increase user familiarity and convenience. Bhattacharjee, ¶145.

Furthermore, POSAs understood that accepting input of the SLA in connection with provisioning new or existing databases beneficially allowed a database to be protected as soon as it is provisioned, since parameters for snapshot and log capturing and retention would already be specified. Bhattacharjee, ¶146. Moreover, it was well-known in database-provisioning UIs both to (1) accept inputs related to retention of snapshots or other backup-related data; and (2) accept SLA-level selection. Bhattacharjee, ¶¶147-149; RDS, 16-17, 187; EX1012, 7; EX1032, 2-3; EX1028, [0046], [0056]-[0058]; EX1029, 9:49-10:12, Fig. 4.

In Sivasubramanian's GUI, POSAs would have been motivated to implement SLA selection via a selectable interface like a drop-down menu, since such interfaces were well-known, including in Sivasubramanian. Bhattacharjee, ¶150; Sivasubramanian, 15:59-60, Fig. 5 (*e.g.*, "Database engine"); EX1017, Fig. 6; RDS, 15. POSAs would have reasonably expected success with such UI

implementations, because SLAs were well-known (*see* above), and because customizing Sivasubramanian’s UIs was within a POSA’s skill (*supra* §VI.A.2). Bhattacharjee, ¶150.

Furthermore, POSAs would have found it obvious to have the SLA specify either one retention period for both snapshots and logs or separate periods for each. POSAs understood there were two options—uniform or non-uniform—and either was an obvious design choice. POSAs would have additionally recognized that uniform periods beneficially simplify snapshot/log management, and POSAs would have reasonably expected success because it was simple and well-known to apply one duration to all snapshots/logs. Bhattacharjee, ¶151; EX1017, [0046]; EX1016, [0009]; RDS, 16, 111-12.

b. UI for protection schedule

As explained below, POSAs would have been motivated to customize Sivasubramanian’s Figure 5 GUI to accept an input defining Zha’s above-discussed “schedule” including a “frequency” and “times during the day” for taking snapshots and backing up transactional/transaction logs. Zha, 11:33-54; *supra* §VI.B. Fields related to database protection in database-provisioning UIs were well-known, and POSAs understood that accepting a protection schedule via Sivasubramanian’s database-provisioning UIs would beneficially allow the database to be protected as soon as it is provisioned, since the schedule would

already be specified. Bhattacharjee, ¶¶152-154; RDS, 16-17, 187 (“backup window” for taking “snapshots”); EX1012, 7 (“Backup Destination”); EX1032, 2-3. POSAs would have reasonably expected success with such UI implementations, because including fields related to database protection in database-provisioning UIs was well-known, as discussed above, and because customizing Sivasubramanian’s UIs was within a POSA’s skill (*supra* §VI.A.2). Bhattacharjee, ¶155.

2. Sivasubramanian+Zha in view of Shekar

POSAs would have found it obvious to further apply the teachings of Shekar to the Sivasubramanian+Zha combination described above, as explained below.

Sivasubramanian states that “*in one embodiment*,” the “data plane” is a “cloud” system. Sivasubramanian, 9:9-13. However, Sivasubramanian does not *require* that the data plane be a cloud. *E.g.*, Sivasubramanian, 24:19-28:56 (Sivasubramanian’s claims not requiring cloud). Rather, POSAs understood that Sivasubramanian is directed towards automating workflows but is flexible regarding the environment where workflows are performed. Sivasubramanian, 4:48-57 (“aspects of [Sivasubramanian’s] various embodiments can be used with any appropriate services available or offered over a network in an electronic environment”), 3:10-25, 9:35-61, 15:45-50, 23:17-24; Bhattacharjee, ¶¶156-157.

One well-known alternative to a cloud-based system was a “hyperconverged infrastructure” (HCI) system. EX1036, [0003] (“Hyper[-]Converged Infrastructure (HCI) combines computing applications and storage applications into a single infrastructure.”); EX1037, 1:13-33; EX1018, [0005]. Furthermore, it was known to deploy databases in hyperconverged systems. Bhattacharjee, ¶158 (citing corroborating EXEX1039, [0242] (discussing “hyperconverged database type management system”); EX1040, [0089]; EX1038, 6:46-53, Fig. 2)

In view of Shekar, POSAs would have been motivated to provide the option of a “data plane” implemented using one or more of Shekar’s “HCIs,” for multiple reasons. First, as noted above, POSAs understood that hyperconverged infrastructure was a known option for implementing a “data plane.” Second, POSAs would have been motivated to implement the data plane as an HCI system as an alternative to a cloud to achieve the known benefits of HCI, such as resource sharing, flexibility, and ease of management. Bhattacharjee, ¶¶159-161; EX1041, [0002]; EX1042, 1:5-10; EX1043, 1:6-20. Third, POSAs would have been specifically motivated to use Shekar’s “HCI[s]” to beneficially achieve the “high performance” of Shekar’s HCIs in a “data plane.” Shekar, [0005]; Bhattacharjee, ¶162. Fourth, Shekar discloses that each HCI includes a “storage array” that provides “storage services” to “host applications” running in the HCI. Shekar, [0035]; *supra* §VI.C. POSA understood that “database[s]” such as those

provisioned by Sivasubramanian (Sivasubramanian, 6:59-56; *supra* §VI.A.1) are applications that use storage services, and thus that would benefit from incorporation of Shekar’s HCI. Bhattacharjee, ¶163; Sivasubramanian, 6:43-58.

The resulting Sivasubramanian+Shekar system includes a “control plane” as taught by Sivasubramanian (*supra* §VI.A.1) that executes workflows in a data plane implemented using one or more of Shekar’s HCIs (*supra* §VI.C). In this system, databases, which are “applications” as noted above, run on “virtual machines” (“VMs”) running on HCIs, as taught in Shekar. Shekar, [0008]-[0010], [0032], [0035], [0038], [0040]; *supra* §VI.C. Additionally, as detailed *infra* [1E] and claim 3, in the resulting system, Sivasubramanian’s graphical and command-line database-provisioning UIs are implemented to accept additional parameters for configuring the VMs. Bhattacharjee, ¶164.

Furthermore, POSAs would have found it obvious to incorporate Zha’s teachings regarding user-defined snapshot and transaction-log collection frequencies, times during the day, and/or retention periods, and also incorporate UI elements to gather such information, into the Sivasubramanian+Shekar system for the same reasons as the Sivasubramanian+Zha combination (*supra* §VI.D). Bhattacharjee, ¶165.

The resulting Sivasubramanian+Zha+Shekar combination introduces two additional aspects to the Sivasubramanian+Zha combination described above: (1)

the data plane is implemented using Shekar’s HCIs, with databases running on VMs on the HCIs; and (2) additional UI elements are added for configuring VMs, as discussed above and detailed in [1E] and claim 3 below. Bhattacharjee, ¶166.

POSAs would reasonably have expected success in the above-described implementation for multiple reasons, as explained below.

Using hyperconverged systems for databases was known in the art, as noted above. Running databases on virtual machines (“VMs”) was also well-known in the art. Bhattacharjee, ¶¶167-168 (citing corroborating EX1065, [0047]; EX1067, [0021]; EX1068, [0007]; EX1069, 3:16-19). Additionally, POSAs understood that a VM emulates a physical computer system; thus, POSAs would have reasonably expected that running databases on VMs on Shekar’s HCIs would not change the functions of the databases in any aspect relevant to Sivasubramanian’s control plane. Bhattacharjee, ¶169 (citing corroborating EX1070, 1:10-17).

Moreover, POSAs understood that Shekar’s “HCI” includes components that are functionally equivalent to components that Sivasubramanian describes as being used to run databases in the data plane. Bhattacharjee, ¶170. Shekar’s HCI includes a “storage array” providing “storage services” to “host applications” running in the HCI. Shekar, [0035]; *supra* §VI.C. Similarly, Sivasubramanian discloses that a “data instance” for a database (*see supra* §VI.A.1), is “allocate[ed]” and “attach[ed]... to” “off-instance persistent storage.”

Sivasubramanian, 6:59-65. As another example, each of Shekar’s “virtual host systems” (also called “VMs”) on which “applications” run includes a “VCPU” (virtual CPU) and “virtual memory (RAM).” Shekar, [0046]. Similarly, Sivasubramanian discloses that “instances” for databases may have an “instance type” that “describes the CPU and memory capacity.” Sivasubramanian, 11:49-52.

Furthermore, Shekar explains that the nodes in an HCI include “one or more network interface cards (NICs)... for exchanging data packages through a network.” Shekar, [0039]. Thus, POSAs understood that Sivasubramanian’s control plane would be able to communicate with the databases running on VMs in the HCIs to execute workflows. Bhattacharjee, ¶¶171-172.

Finally, implementing Sivasubramanian’s UI to accept additional parameters was within a POSA’s skill. *Supra* §VI.A.2.

E. Claim-by-Claim Analysis

1. Claim 1

a. [1Pre]: “A system comprising:”

Sivasubramanian discloses a “**system** for controlling... a configuration aspect of at least one data instance in [a] data environment.” Sivasubramanian, claims 20, 24; *see also* 2:17-18, 4:48-53. As explained below, Sivasubramanian’s *system*, in view of Zha’s and Shekar’s teachings (*see supra* §VI.D), *comprises* claim 1’s remaining elements. Bhattacharjee, ¶173.

b. [1A]: “a memory storing computer-readable instructions in a virtual computing system; and”

Sivasubramanian’s invention can be implemented with a “computing device” including “*computer readable media*” “*storing*” “*computer readable instructions*,” examples of such media include various “*memory*” types.

Sivasubramanian, 22:33-37, 23:42-48, 23:61-24:8; Bhattacharjee, ¶174.

Sivasubramanian also says the “control plane” includes “computer-executable software” (*i.e. instructions*) and Sivasubramanian includes claims reciting a “control environment” operating “under control of” a “computer system[] configured with “executable *instructions*.” Sivasubramanian, 5:35-39, 24:20-24, 25:4-8, 26:23-26; Bhattacharjee, ¶174. Thus, Sivasubramanian discloses implementing its “control plane” using *a memory storing computer-readable instructions*. Bhattacharjee, ¶174.

As explained *supra* §VI.D.2, in Sivasubramanian+Zha+Shekar, databases run on virtual machines in Shekar’s HCI; therefore, Sivasubramanian+Zha+Shekar’s *system* is a *virtual computing system*. EX1001, 1:10-12 (“[v]irtual computing systems include one or more host machines running one or more virtual machines and other entities”). Thus, in Sivasubramanian+Zha+Shekar, the *memory storing the computer-readable instructions* that implement Sivasubramanian’s “control plane” is in a *virtual computing system*. Bhattacharjee, ¶175.

Further, if the claim requires the *memory* storing the *computer-readable instructions* to itself be a virtual component, virtual machines were well-known in the art and POSAs would have found it obvious for the “control plane” to execute within such a virtual machine, with its instructions stored and executed from a virtual memory component. Bhattacharjee, ¶176; Sivasubramanian, 16:60-63; Shekar, [0046] (describing “virtual host system” (*i.e.*, a virtual machine) including “virtual memory”).

c. [1B]: “a processor executing the computer-readable instructions to:”

Sivasubramanian teaches *a processor executing* the aforementioned *computer-readable instructions* implementing the “control plane.” Sivasubramanian’s above-discussed “computer-readable instructions” for the control plane are intended to be executed by a processor. Sivasubramanian, 27:18-35 (claim 20 reciting “system for controlling a data environment using a control environment” comprising “processor” and “instructions that, when executed by” the “processor” “cause the system” to perform actions), 5:35-39 (control plane includes “computer-executable software”); Bhattacharjee, ¶177.

As explained below, all steps recited in [1C]-[1K] are performed as part of provisioning a database using the control plane’s database-provisioning UIs; thus, since the *processor* executes the control-plane instructions as discussed above, *the processor* performs the claim’s recited tasks. Bhattacharjee, ¶178.

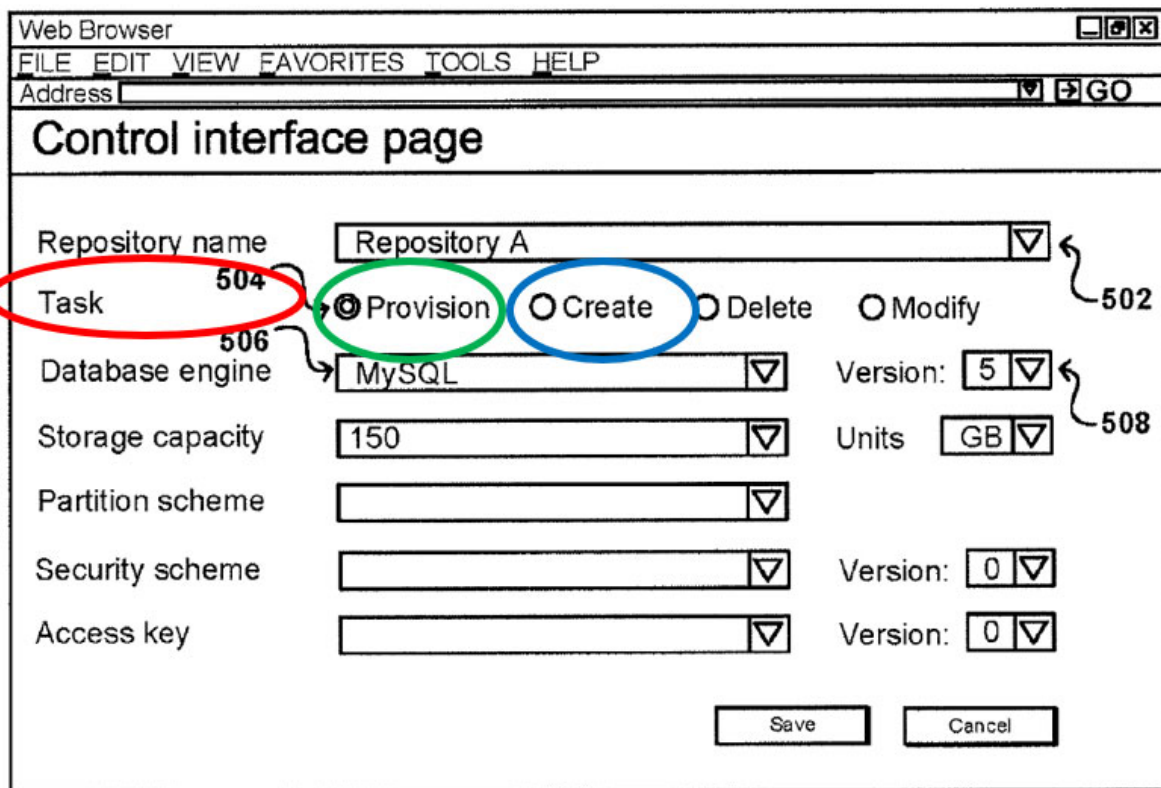
d. [1C]: “present a dashboard to receive a user input in a graphical user [interface]⁴ (GUI) to provision a source database;”

As explained *supra* §VI.A, and more specifically §§VI.A.1, VI.A.1.a, Sivasubramanian discloses starting the process for *provisioning* a *database* in response to a *user input*. Sivasubramanian’s Figure 5 “illustrates an example of a *display*” (*i.e.*, a *graphical user interface*) “to make calls into the control plane” when the user “*wants to provision a new data instance*” (*i.e.*, a *database*, *supra* §VI.A.1). Sivasubramanian, 12:38-40, 12:52-55. As explained *supra* §VI.A.1.a.i, control-plane users could either: (1) select the “Provision” option to *provision* an existing *database*; or (2) select the “Create” option to *provision* a newly-created database (as discussed *supra* §VI.A.1). Either selection made by the user through the display is therefore a *user input in a GUI to provision a database*.

Bhattacharjee, ¶180.

Figure 5 below shows Sivasubramanian’s *GUI*:

⁴ [1C] recites “graphical user *interlace* (GUI),” For present purposes, Petitioner interprets “interlace” as a typographical error. Bhattacharjee, ¶179; EX1001, 12:31-32.



Sivasubramanian, Fig. 5 (annotated)

POSAs understood that a GUI like Sivasubramanian's Figure 5 that lets a user provision and manage databases was commonly called a "*dashboard*."

Bhattacharjee, ¶181-182 (citing corroborating EX1062, [0123]-[0144], [0502]-[0507]; EX1063, [0297]-[0307], Fig. 8A). Furthermore, like the '157 patent's "dashboard," Sivasubramanian's Figure 5 GUI "forms the front end of the database system" that the user can interact with. EX1001, 11:49-52; Sivasubramanian, 5:58-62; Bhattacharjee, ¶183.

The '157 patent describes a "*source database*" as a database from which snapshots are taken to create backups. See, e.g., EX1001, 4:46-57, 5:20-23. As

discussed *infra* [11], both the “Provision” and “Create” tasks in the GUI set up the database being provisioned to have snapshots captured from it to create backups, according to either implementation of backing up data discussed *supra* §VI.D; thus, in Sivasubramanian+Zha+Shekar, the *database* that is *provisioned* is a *source database*. Bhattacharjee, ¶184; Zha, 1:66-2:6 (referring to backed-up database as “*source database*”).

Sivasubramanian+Zha+Shekar meets [1C] for an additional, alternate reason. Sivasubramanian describes Figure 5’s *GUI* as an “interface page” that a “*user*... will access... through the Internet.” Sivasubramanian, 12:48-52. POSAs understood that the *user* “access[es]” the page by making a “request[]” the control plane “*receives*.” Sivasubramanian, 5:35-41; Bhattacharjee, ¶185. This “*user*” “access” request is *also* a *user input to provision a database*, because this request invokes Figure 5’s database-*provisioning* UI (presenting the “Create” or “Provision” tasks, as discussed above). Bhattacharjee, ¶185; EX1001, 13:2-5. POSAs would have found it obvious to implement Sivasubramanian+Zha+Shekar so that this *input* is received on a different screen that is itself a *dashboard in a GUI*, as explained below. Such dashboards from which users can select options to do tasks like creating or provisioning were well-known as discussed above. POSAs would have been motivated to implement Sivasubramanian+Zha+Shekar’s GUI this way to beneficially provide a familiar user interface to the user, and

would reasonably have expected success because such UI implementations were within a POSA's skill. Bhattacharjee, ¶185; *supra* §VI.A.2.

- e. **[1D]: “upon receiving the user input, present options in a first user interface of the GUI to receive a first selection of a database engine type;”**

Sivasubramanian's Fig. 5 (annotated below) shows a *first selection* of a type of “***Database engine***” (element 506), *i.e.*, a *database engine type*, which in the example of Fig. 5 is “MySQL” (a known type of database engine); POSAs understood that the selection is from *options* that are *presented* in the “Database engine” drop-down menu, which is a *first user interface of the GUI*.

Sivasubramanian, 2:44-47, 10:37-40 (referring to “the type of database engine being used”); EX1001, 14:14-25, 24:9-20; EX1101, 1; Bhattacharjee, ¶¶186-187.

The screenshot shows a web browser window titled 'Web Browser' with a menu bar (FILE, EDIT, VIEW, FAVORITES, TOOLS, HELP) and an address bar. The main content area is titled 'Control interface page'. It contains several form fields and controls:

- Repository name:** A dropdown menu showing 'Repository A'.
- Task:** Four radio buttons labeled 'Provision' (selected), 'Create', 'Delete', and 'Modify'.
- Database engine:** A dropdown menu showing 'MySQL'. This field is circled in red and annotated with '506'.
- Version:** A numeric input field showing '5'.
- Storage capacity:** A numeric input field showing '150'.
- Units:** A dropdown menu showing 'GB'.
- Partition scheme:** An empty dropdown menu.
- Security scheme:** An empty dropdown menu.
- Access key:** An empty dropdown menu.
- Version:** Two numeric input fields, both showing '0'.
- Buttons:** 'Save' and 'Cancel' buttons at the bottom right.

Annotations with arrows point to specific elements:

- '504' points to the 'Provision' radio button.
- '502' points to the 'Repository A' dropdown.
- '506' points to the 'MySQL' dropdown (circled in red).
- '508' points to the 'Version: 5' input field.

Sivasubramanian, Detail of FIG. 5 (annotated)

While Figure 5 illustrates a specific example for the “Provision” task (selected under element 504), POSAs understood that the GUI would similarly display a field for “database engine” *selection* when the “Create” task is selected under element 504 (*i.e., upon receiving the user input of [1C]*, as discussed below). *Supra* §VI.A.1.a.i; Bhattacharjee, ¶188. POSAs also understood that a dropdown menu UI element, such as “database engine” element 506 in Figure 5, *receives a user selection* of one of the *options* presented (e.g., a particular database engine type, such as “MySQL”), and Sivasubramanian discusses the control plane performing workflows or actions for a user-specified “engine;” thus, POSAs

understood that Sivasubramanian’s GUI *receive[s] a first selection of a database engine type*. Bhattacharjee, ¶189; Sivasubramanian, 5:59-62 (referencing “[w]eb service frontend features such as... validating user input...”), 12:55-57 (Fig. 5 UI lets user “enter information”), 6:39-41, 10:37-40, 12:58-61, 19:25-31.

If [1D] requires the *options* vary based on [1C]’s *user input*, POSAs understood, or at least would have found obvious, that “database engine” selection field (element 506) in Sivasubramanian’s FIG. 5 was populated with different database engine types depending on whether the “Create” or “Provision” task was selected by the user (*i.e., upon receiving the user input* of [1C]). Sivasubramanian, 2:44-47, 10:37-40; Bhattacharjee, ¶190. For example, selecting the “Create” task would populate element 506 with options for all database engine types supported by Sivasubramanian’s system, whereas selecting the “Provision” task, which is an operation on previously-created databases, would only populate element 506 with options for database engine types that were already in use in Sivasubramanian’s system. Sivasubramanian, 2:44-47; Bhattacharjee, ¶190. POSAs understood that populating the element 506 only with database engine types for which the selected task was valid would prevent the submissions of invalid tasks to Sivasubramanian’s control plane, which would beneficially lead to better system efficiency and improved user input validation by the control plane. Bhattacharjee, ¶191. Sivasubramanian, 5:58-62 (describing one function of the control plane as

“validating user input”). This would also beneficially increase user convenience and was within a POSA’s skill. Bhattacharjee, ¶191; *supra* §VI.A.2.

- f. [1E]: “upon receiving the first selection of the database engine type, present a plurality of selectable options to be selected via at least one user interface of the GUI, the plurality of selectable options comprising at least a plurality of:”

Sivasubramanian teaches *presenting a plurality of selectable options to be selected via at least one user interface of the GUI*, as shown in Fig. 5:

The screenshot shows a web browser window titled "Web Browser" with a menu bar (FILE, EDIT, VIEW, FAVORITES, TOOLS, HELP) and an address bar. The main content area is titled "Control interface page". It contains several form fields and controls:

- Repository name:** A text field containing "Repository A" with a dropdown arrow (annotated 502).
- Task:** A group of radio buttons with labels "Provision" (selected), "Create", "Delete", and "Modify" (annotated 504).
- Database engine:** A text field containing "MySQL" with a dropdown arrow (annotated 506).
- Version:** A text field containing "5" with a dropdown arrow (annotated 508).
- Storage capacity:** A text field containing "150" with a dropdown arrow.
- Units:** A text field containing "GB" with a dropdown arrow (annotated 508).
- Partition scheme:** A text field with a dropdown arrow.
- Security scheme:** A text field with a dropdown arrow.
- Access key:** A text field with a dropdown arrow.
- Version (bottom):** Two text fields, each containing "0" with a dropdown arrow.
- Buttons:** "Save" and "Cancel" buttons at the bottom right.

A red dashed box highlights the "Storage capacity", "Partition scheme", "Security scheme", and "Access key" fields. Arrows point from the annotations 502, 504, 506, and 508 to their respective fields.

Sivasubramanian, Detail of FIG. 5 (annotated)

As previously discussed (*supra* §§VI.A.2, VI.D), POSAs would have been motivated and found it obvious in Sivasubramanian+Zha+Shekar to present

alternative and/or additional selectable options in Sivasubramanian’s GUI beyond the specific examples of options illustrated in FIG. 5, such as those discussed in the sections below for [1F]-[1I]. Bhattacharjee, ¶¶192-193.

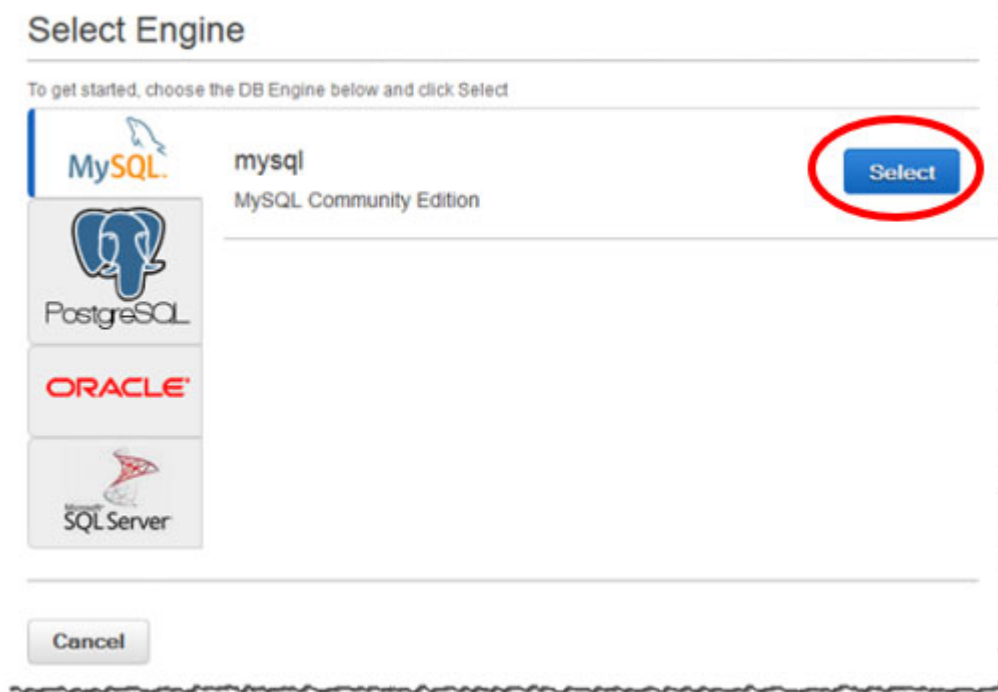
Additionally, in Sivasubramanian+Zha+Shekar, the “*the plurality of selectable options*” recited in [1F]-[1I] (discussed *infra* §§VI.E.1.g-j) are presented “*upon receiving the first selection of the database engine type*” (discussed *supra* [1D]). POSAs understood, or at least would have found obvious, that the fields of the Sivasubramanian+Zha+Shekar UI corresponding to the various *selectable options* recited in [1F]-[1I] would be populated with appropriate sets of options for the particular database engine type selected in [1D], because POSAs understood that the appropriate sets of options for the items recited in [1F]-[1I] may vary based on the selected engine type. Sivasubramanian, 2:44-47 (mentioning various engine types), 10:37-40 (“different tasks can be selected for the workflow based upon factors such as the type of action requested and the type of database engine being used”); Bhattacharjee, ¶194. For example, selecting “MySQL” as the database engine type in element 506 of Sivasubramanian’s FIG. 5 UI would cause the various UI elements recited in [1F]-[1I] to populate with: *first profile[s] for creating a source database virtual machine compatible with MySQL databases (infra §VI.E.1.g ([1F]))*; *second profile[s] for creating the source database including parameters specific to MySQL databases (infra §VI.E.1.h ([1G]))*;

Service Level Agreement[s] (“*SLA[s]*”) that have been configured for use with MySQL databases (*infra* §VI.E.1.i ([1H])); and *frequenc[ies]* of capturing a snapshot and transactional log supported by the MySQL database engine (*infra* §VI.E.1.j ([1I])). Sivasubramanian, 2:44-47, 10:37-40; Bhattacharjee, ¶194.

POSAs understood that populating the elements of Sivasubramanian’s UI only with options supported by or compatible with the selected database engine type would prevent the submissions of invalid configuration tasks to Sivasubramanian’s control plane (*e.g.*, requests to create a MySQL database with unsupported or invalid options), which would lead to better system efficiency and improved user input validation by the control plane. Bhattacharjee, ¶195; Sivasubramanian, 5:58-62 (describing one function of the control plane as “validating user input”). This would also beneficially increase user convenience by restricting the user to valid configuration options for the selected engine type, and would also further Sivasubramanian’s stated goal for the “control plane to act as a virtual database administrator (DBA),” including “***determine[ing] whether the configuration is valid***” when creating or provisioning a database. Sivasubramanian, 9:45-54; Bhattacharjee, ¶195.

Furthermore, POSAs understood that presenting a sequential UI (either on one screen, with options for later items changing based on selection of options for previous items, or multiple screens) that guided users through a multi-phase task

and presented options based on previous selections—often referred to as a “wizard”—was well known in the art, including in database creation and provisioning UIs. Bhattacharjee, ¶196 (citing corroborating EX1108, [0013]; EX1109, [0040]; RDS, 12-17 (describing UI with “Select Engine” page followed by “Specify DB [database] Details” and “Advanced Settings” pages, shown below)). POSAs would reasonably have expected success with such an implementation because sequential UIs were well-known and because implementing such UIs was within a POSA’s skill (*supra* §VI.A.2). Bhattacharjee, ¶196.



RDS, 13 (annotated)

Specify DB Details

Instance Specifications

DB Engine: mysql

License Model: general-public-license

DB Engine Version: 5.6.19a

Review the [Known Issues/Limitations](#) to learn about potential compatibility issues with specific database versions.

DB Instance Class: - Select One -

Multi-AZ Deployment: - Select One -

Storage Type: - Select One -

Allocated Storage*: 5 GB

Provisioning less than 100 GB of General Purpose (SSD) storage for high throughput workloads could result in higher latencies upon exhaustion of the initial General Purpose (SSD) IO credit balance. [Click here](#) for more details.

Settings

DB Instance Identifier*:

Master Username*:

Master Password*:

Confirm Password*:

* Required

[Cancel](#) [Previous](#) [Next Step](#)

RDS, 15 (annotated)

Configure Advanced Settings

Network & Security

This instance will be created with the new Certificate Authority rds-ca-2015. If you are using SSL to connect to this instance, you should use the [new certificate bundle](#). Learn more [here](#).

VPC* Default VPC (vpc-...)

Subnet Group default

Publicly Accessible Yes

Availability Zone No Preference

VPC Security Group(s) Create new Security Group
default (VPC)

Database Options

Database Name

Note: If no database name is specified then no initial MySQL database will be created on the DB instance.

Database Port 3306

DB Parameter Group default:mysql5.6

Option Group default:mysql-5-6

Copy Tags To Snapshots ☐

Enable Encryption No

Backup

Please note that automated backups are currently supported for InnoDB storage engine only. If you are using MyISAM, refer to [detail here](#).

Backup Retention Period 7 days

Backup Window No Preference

Monitoring

Enable Enhanced Monitoring No

Maintenance

Auto Minor Version Upgrade Yes

Maintenance Window No Preference

* Required

Cancel Previous Launch DB Instance

RDS, 17

Element [1E] further recites that “*the plurality of selectable options*” presented “*compris[e] at least a plurality of*” the features recited in [1F]-[1I]. As

described *infra* §§VI.E.1.g-j, Sivasubramanian+Zha+Shekar meets all of these elements. Moreover, if limitation [1G] is interpreted to require only two or more of these features, due to the language “*a plurality of*,” Sivasubramanian+Zha+Shekar also meets [1G]. Bhattacharjee, ¶197.

g. [1F]: “a second selection of a first profile for creating a source database virtual machine;”

The ’157 patent describes a “database virtual machine” as a VM that “host[s]” a database, and describes a “source database virtual machine” as a virtual machine “associated with” the “storage” used by a source database. *See, e.g.*, EX1001, 3:32-49, 14:59-61; Bhattacharjee, ¶198.

In Sivasubramanian+Zha+Shekar, databases can run on *virtual machines* (VMs) on hyperconverged infrastructures (HCIs), as discussed *supra* §VI.D. Shekar refers to VMs as “*host[s]*” for applications, and each VM includes a “storage array manager” that allows “applications” to access the “storage array.” *E.g.*, Shekar, [0046]-[0047], Fig. 2B. In Sivasubramanian+Zha+Shekar, the databases running on Shekar’s VMs use Shekar’s storage array for storage. *Supra* §VI.D.2. Thus, in Sivasubramanian+Zha+Shekar, a virtual machine on which a database runs is a *source database virtual machine*. Bhattacharjee, ¶¶199-200.

As Shekar explains, POSAs knew that (1) VMs have properties including “operating system” and amount of “memory (RAM),” and (2) VMs can be “configured.” Shekar, [0038], [0046], Fig. 2B; Bhattacharjee, ¶201. Furthermore,

it was known to customize the properties of VMs, including VMs on which databases ran. Bhattacharjee, ¶201 (citing corroborating EX1070, 1:11-15 (VMs “can be customized” with “computing resources” including “RAM,” and “operating system”); EX1067, [0029]-[0035] (describing “[VM] templates” for VMs running databases); EX1071, [0158]-[0159]; EX1072, [0066]; EX1069, 1:19-25, 7:49-8:2). Thus, in Sivasubramanian+Zha+Shekar, where a database runs on a VM in Shekar’s HCI (*supra* §VI.D.2), POSAs would have found it obvious to implement Sivasubramanian’s database-provisioning GUI (*supra* §VI.A.1.a) to *present a selectable option comprising a second selection of a set of values for VM properties (together forming a first profile), including an OS and an amount of RAM, to customize the VM. The first profile would be used for creating a source database virtual machine upon which the source database (described supra [1C]) would run as an “application” that “resid[es]” on the source database virtual machine, as taught by Shekar. Shekar, [0008], [0038], [0035], [0046], Fig. 2B (showing “applications 155” inside “VM”); Bhattacharjee, ¶¶202-203.*

POSAs would have been motivated to implement this functionality to beneficially give control-plane users flexibility to customize VMs based on the database’s needs, and such an implementation would have been within a POSA’s skill. *Supra* §VI.A.2; Bhattacharjee, ¶204.

h. [1G]: “a third selection of a second profile for creating the source database;”

Sivasubramanian’s UIs accept *selection* of parameters related to *creating the source database* (discussed *supra* [1C]), including “Partition scheme,” “Security scheme,” and “password.” Sivasubramanian, 13:1-5, Fig. 5. *Selection* of these parameters collectively constitute a *third selection of a second profile for creating the source database*. Bhattacharjee, ¶205.

The screenshot shows a web browser window titled "Web Browser" with a menu bar (FILE, EDIT, VIEW, FAVORITES, TOOLS, HELP) and an address bar. The main content area is titled "Control interface page". It contains several form elements: "Repository name" (dropdown menu with "Repository A" selected), "Task" (radio buttons for Provision, Create, Delete, Modify), "Database engine" (dropdown menu with "MySQL" selected), "Storage capacity" (text input with "150"), "Units" (dropdown menu with "GB" selected), "Partition scheme" (dropdown menu), "Security scheme" (dropdown menu), and "Access key" (text input). There are also "Version:" labels next to the "Database engine", "Units", "Security scheme", and "Access key" fields, each with a small dropdown menu. At the bottom, there are "Save" and "Cancel" buttons. Annotations with numbers 504, 506, 502, and 508 point to specific elements: 504 points to the "Repository name" dropdown, 506 points to the "Database engine" dropdown, 502 points to the "Task" radio buttons, and 508 points to the "Units" dropdown.

FIG. 5

Sivasubramanian, Fig. 5

If [1G] is interpreted to require the *second profile* be a single predefined set of values, this would have been obvious. Defining sets of parameters values, which were often called “profiles” or “templates,” was well-known. Bhattacharjee, ¶206 (citing corroborating EX1073, Abstract, 1:31-34; RDS, 412; EX1067, [0029];

EX1062, [0734], Fig. 9; EX1072, [0066]; EX1074, [0002]; EX1075, [0028]; EX1076, [0022]; EX1077, [0050]).

It was likewise well-known that allowing selection of a predefined “profile”/“template” beneficially increased user convenience by reducing the need for users to choose and separately input parameter values. Bhattacharjee, ¶207 (citing corroborating EX1078, 1-2; EX1079, [0003]; EX1031, 1:16-29; EX1080, [0225]-[0226], Fig. 16A; EX1081, [0105]). Implementing Sivasubramanian’s UI to accept such “profile”/“templates” was within a POSA’s skill. *Supra* §VI.A.2.

i. [1H]: “a fourth selection of a Service Level Agreement (“SLA”); and”

The ’157 patent says “[a]n *SLA is an agreement*... that outlines, among other things, the protection scope of the database. The protection scope defines for how long data from the database being created or registered is retained.” EX1001, 19:12-23. Sivasubramanian+Zha+Shekar’s database-provisioning GUI includes a UI element enabling *selection* of an *SLA* that specifies snapshot and transaction-log retention periods. Bhattacharjee, ¶208; Zha, 11:33-44; *supra* §§VI.D.1.a, VI.D.2.

Thus, in Sivasubramanian+Zha+Shekar, Sivasubramanian’s GUI is operable to *present a selectable option comprising a fourth selection of a Service Level Agreement (“SLA”)*. Bhattacharjee, ¶209.

j. [1I]: “a fifth selection of a frequency of capturing a snapshot and transactional log;”

In Sivasubramanian+Zha+Shekar, Sivasubramanian’s database-provisioning GUI accepts a user-specified “schedule” as Zha teaches. *Supra* §§VI.D.1.b, VI.D.2; Zha, 11:33-44. This “schedule” includes (1) a “*frequency*” for “retriev[ing] PIT copies,” *i.e.*, *capturing snapshots*, and (2) a “*frequency*” for “retriev[ing] updates to online transaction logs,” *i.e.*, *capturing transactional logs*. Zha, 11:33-44, 15:21-24; Bhattacharjee, ¶210; EX1001, 5:4-8; EX1009, 277.

The ’157 Patent describes that the frequency of capturing snapshots and the frequency of capturing transactional logs should be different due to the different impacts on the operation of the database of the two tasks. EX1001, 5:36-37 (“The frequency of capturing transactional logs may be higher than the frequency of capturing snapshots”), 5:41-44, 5:46. Therefore, the user-specified “schedule” selectable through Sivasubramanian’s database-provisioning GUI is a *selectable option comprising a fifth selection of a frequency of capturing a snapshot and transactional log*. Bhattacharjee, ¶211.

k. [1J]: “upon receiving the selections of the database engine type and the at least plurality of the first profile, the second profile, the SLA, and the frequency of capturing the snapshot and transactional log, create the source database and associate the created

source database with the source database virtual machine; and”

In Sivasubramanian+Zha+Shekar, the various *selections* recited in [1J] are received via the GUI for database creation. *Supra* [1E]-[1I]. POSAs would have understood that the purpose of accepting these selections is to use them to “*create*” a database based on these selections. Sivasubramanian, 11:47-52 (“customers can *create* a data store using a ‘Create*Database*’ or similar API. The user can call a Web service to specify any desired values for an instance type ..., storage size, repository name, port, and other such values.”), 12:38-13:9 (describing creating database using Fig. 5 GUI); *supra* §VI.D (explaining “create database” workflow’s use of parameters specified in SLA and protection schedule in Sivasubramanian+Zha+Shekar); Bhattacharjee, ¶212.

Furthermore, POSAs would have understood that the processor (*supra* [1B]) *creates the source database and associates it with the source database virtual machine* described by the *first profile* after creation. In Sivasubramanian+Zha+Shekar, a database runs as an “application” on a VM running on an HCI, as Shekar teaches. Shekar, [0008]; *supra* §VI.D.2. In Shekar, “applications” such as databases “resid[e] on” VMs on Shekar’s HCIs; thus, in Sivasubramanian+Zha+Shekar, the source database resides on the created virtual machine, and is thus *associated* with it. Shekar, [0038], [0035], [0046], Fig. 2B (showing “applications 155” inside “VM”); Bhattacharjee, ¶213.

As discussed *supra* [1B], the *processor* executing the control plane instructions performs the tasks recited in claim 1; thus, the processor *creates* and *associates* the *source database* as recited. Bhattacharjee, ¶213.

1. [1K]: “display a status of the created source database on the dashboard.”

Sivasubramanian teaches that, after creating a database, a request issued to the control plane using “a ‘DescribeDatabase’ or similar API” can be utilized “to poll on the *status* of the repository” (*i.e., the created source database*) “to determine the state of the repository, such as whether the repository state is provisioned.” Sivasubramanian, 11:52-56. Sivasubramanian teaches that control-plane requests can be issued from, and results displayed, in a webpage such as the FIG. 5 GUI (*i.e., the dashboard, supra* [1C]). Sivasubramanian, 12:49-55; Bhattacharjee, ¶214. POSAs would have found it obvious to have the GUI automatically display the status upon creation of the database, because POSAs understood that users would want to know if the database creation operation was successful, and such a UI implementation was within a POSA’s skill. *Supra* §VI.A.2; Bhattacharjee, ¶215.

2. Claim 2

Sivasubramanian discloses a “DescribeDatabase” command for “poll[ing]” the “repository[’s]” “status,” allowing changes in the list of provisioned source databases to be observed over time; Sivasubramanian also mentions a

“DescribeDatabases” (plural) command. Sivasubramanian, 11:52-55, 13:20-21; Bhattacharjee, ¶216.

Sivasubramanian teaches providing users the same functionality of the command line and/or APIs (such as the “DescribeDatabases” command) through a GUI dashboard, like Fig. 5’s example. *Supra* §VI.A. Thus, POSAs would have found it obvious for Sivasubramanian+Zha+Shekar’s GUI *dashboard* (*see supra* [1C]) to *comprise a dynamically changing home page displaying a list of all source databases in the virtual computing system* using data from Sivasubramanian’s “DescribeDatabase” or “DescribeDatabases” command. *Supra* §VI.A.2; Bhattacharjee, ¶¶217-218.

Further, POSAs would have found it obvious for the *dynamically changing home page* to *comprise a plurality of cells with a first cell* displaying the database list; such UI layouts were well-known, therefore such an implementation would be an application of a known technique to Sivasubramanian+Zha+Shekar’s GUI dashboard to achieve predictable results. Bhattacharjee, ¶219 (citing corroborating EX1062, [2008], FIG. 24 (showing UI with plurality of cells, with one cell listing all databases in the system); EX1063, Fig. 8C; EX1067, Fig. 4). Furthermore, it was within a POSA’s skill to customize Sivasubramanian’s GUI. *Supra* §VI.A.2.

3. Claim 3

a. “software profile”

A “*software profile*” defines the software and operating system [OS] parameters for the database engine type... selected” during the database “provisioning” process. EX1001, 17:13-19, 15:52-57; Bhattacharjee, ¶220. Sivasubramanian+Zha+Shekar’s *first profile* comprises the OS for the VM that the database of the selected engine type runs on. *Supra* [1F]. Additionally, Sivasubramanian discloses specifying “version information” for the selected engine type in Figure’s 5 GUI. Sivasubramanian, 12:57-13:5, Fig. 5; Bhattacharjee, ¶220.

Web Browser
FILE EDIT VIEW FAVORITES TOOLS HELP
Address GO

Control interface page

Repository name

Task ☒ Provision ☐ Create ☐ Delete ☐ Modify

Database engine Version:

Storage capacity Units

Partition scheme

Security scheme Version:

Access key Version:

FIG. 5

Sivasubramanian, Fig. 5

POSAs understood the portion of Sivasubramanian's FIG. 5 GUI that collects the various profiles claim 3 recites to be a *second user interface* distinct from the *first user interface* of [1D] that receives the database engine type. Bhattacharjee, ¶221. If the claim requires the *second user interface* to be on a separate UI screen, adding a separate screen as part of a sequential UI would have been obvious for the same reasons discussed *supra* [1E]. Thus, *to receive the selection of the first profile, the second user interface comprises fields for receiving selection of the OS and the database version, which together comprise a software profile.* Bhattacharjee, ¶221.

If the claim requires the “software profile” be a predefined set of values, this would have been obvious. Defining sets of parameters values (often called “profiles” or “templates”) was well-known. *Supra* [1G]; Bhattacharjee ¶222, 206 (citing corroborating exhibits discussed *supra* [1G]). It was likewise well-known that allowing selection of a predefined “profile”/“template” beneficially increased user convenience by reducing the need to choose and separately input parameter values. *Supra* [1G]; Bhattacharjee, ¶223, 207 (citing corroborating exhibits discussed *supra* [1G]). POSAs also understood that certain versions of applications like database engines were compatible with, or optimized for, specific OSs or versions thereof. Bhattacharjee, ¶224 (citing corroborating EX1082, [0027]; EX1083, [0013]; EX1084, [0012]; EX1085, [0031]; EX1087, [0015];

EX1088, 1). Thus, in Sivasubramanian+Zha+Shekar, POSAs would have found it obvious to implement Sivasubramanian's database creation and provisioning UIs to allow selection of a predefined OS and supported database engine version, *i.e.*, a *software profile*; this would beneficially increase user convenience and was within a POSA's skill. Bhattacharjee, ¶225; *supra* §VI.A.2

b. "compute profile"

A "compute profile defines the size/configuration of [a] source database VM," *e.g.*, "the number of vCPUs [virtual CPUs]" or the "memory capacity." EX1001, 17:42-46.

In Sivasubramanian+Zha+Shekar, the *first profile* comprises the amount of memory for the VM. *Supra* [1F]. Additionally, each VM has a "VCPU." Shekar, [0046]. In Sivasubramanian+Zha+Shekar, where the user may customize the source database's VM, POSAs would have found it obvious to implement Sivasubramanian's database creation and provisioning UIs so the *first profile* received for configuring the VM (*supra* [1F]) includes a number of VCPUs and an amount of memory. Bhattacharjee, ¶¶226-227. Both were well-known configuration options, and such an implementation was within a POSA's skill. *Supra* §VI.A.2; Bhattacharjee, ¶227 (citing corroborating RDS, 14, 104-105; EX1044, [0032]). If the claim requires a predefined set of these parameter values, this was a known UI feature that would have been obvious for the convenience

reasons discussed above for software profiles, and such an implementation was within a POSA's skill. Bhattacharjee, ¶228 (citing corroborating RDS, 14, 104-105; EX1044, [0032]); *supra* §VI.A.2.

POSAs understood the fields for selection of the *compute profile* are part of the *second user interface* for the same reasons discussed above for the *software profile*. Bhattacharjee, ¶229.

c. “network profile”

Sivasubramanian's “example ‘create database’ workflow” includes “allocating and attaching a DNS address or other address, port, interface, or identifier,” and “a user is provided with the DNS address and a port address... to access the [database] instance.” Sivasubramanian, 6:59-67, 11:40-52; Bhattacharjee, ¶230. Additionally, Sivasubramanian's exemplary invocation of “rds-create-database” uses a “port” parameter. Sivasubramanian, 13:1-5.

Thus, POSAs would have been motivated to implement Sivasubramanian's database-provisioning GUI in Sivasubramanian+Zha+Shekar so that the *first profile* received for configuring the source database's VM (*supra* [1F]) includes the DNS address and port; such an implementation would beneficially enable access to the provisioned database by an entity “provide[d] with” that information, and was within a POSA's skill. Sivasubramanian, 6:59-67; Bhattacharjee, ¶231; *supra* §VI.A.2.

A “*network profile*” in the ’157 patent “identifies the network location of the database being created... to facilitate access to the database after creation.” EX1001, 17:31-33. POSAs understood that in Sivasubramanian, the “DNS address” identifies a database’s network “location.” Sivasubramanian, 2:40-44 (referencing “the *DNS (domain name system) address or other location information*” for a database instance); Bhattacharjee, ¶232; EX1033, [0003]. POSAs also understood that the “port” identifies the network location of a database, because an external entity “use[s]” a “port address” to “access” the database. Sivasubramanian, 6:65-67; Bhattacharjee, ¶233. Thus, the DNS address and port together constitute a *network profile*. Bhattacharjee, ¶233.

If the claim requires a *network profile* to be a predefined set of database-access-related parameter values, this was a known UI feature that would have been obvious for the convenience reasons discussed above for software profiles. RDS, 412; Bhattacharjee, ¶234.

POSAs understood the fields for selection of the *network profile* are part of the *second user interface* for the same reasons discussed above for the *software profile*. Bhattacharjee, ¶234.

4. Claim 4

In Sivasubramanian+Zha+Shekar, databases can run on *virtual machines* (VMs) on hyperconverged infrastructures (HCIs). *Supra* §VI.D.2, [1F]. As

Shekar explains, POSAs knew that VMs can be “configured.” Shekar, [0038].

Furthermore, it was known to customize the properties of VMs, including VMs on which databases ran, as discussed *supra* [1F]. Bhattacharjee, ¶¶235-236, 201 (citing corroborating exhibits discussed *supra* [1F]).

POSAs understood that assigning a name and description to virtual machines in a system such as Sivasubramanian+Zha+Shekar’s would assist in system management by allowing users to differentiate between different virtual machines running in the system, and therefore would have found it obvious for Sivasubramanian+Zha+Shekar’s GUI to include fields for *receiving selection of a name and description for the source database virtual machine* as part of the *second user interface* that receives the *first profile* that is for creating the VM as discussed *supra* [1F]. Bhattacharjee, ¶237 (citing corroborating EX1062, [2354]-[2362], Fig. 56 (receiving a selection of a name and description of a virtual machine through a GUI); EX1067, [0030] (administrator providing a “name” and “description” for a virtual machine instance). As previously discussed, it was within a POSA’s skill to customize Sivasubramanian’s GUI. *Supra* §VI.A.2.

5. Claim 5

The ’157 patent uses “*duration*” in the context of snapshots to refer to how long a captured snapshot is stored. EX1001, 20:17-24; Bhattacharjee, ¶238. The patent describes the “daily” snapshot as the snapshot taken at a “designated” daily

time; this snapshot is stored for a longer duration than other snapshots taken that day (e.g., 90 days vs. 30). EX1001, 19:61-67, 21:4-12 (“GOLD” SLA).

“Weekly,” “monthly,” and “quarterly” snapshots can be similarly designated.

EX1001, 20:1-48; Bhattacharjee, ¶238.

In Sivasubramanian+Zha+Shekar, Sivasubramanian’s GUI accepts a “schedule” specifying “times during the day” for the “point in time” copies (*snapshots*) and the “retention period” (*duration*) for such copies, as Zha teaches. Zha, 11:33-44, claim 11; *supra* §VI.D. The portion of Sivasubramanian’s GUI that accepts the “schedule” and “retention period” is a *third user interface*.

Bhattacharjee, ¶239. If the claim requires the *third user interface* to be on a separate UI screen, adding a separate screen as part of a sequential UI would have been obvious for the same reasons discussed *supra* [1E]. Bhattacharjee, ¶240.

Thus, Sivasubramanian+Zha+Shekar’s GUI *receives selection of a duration for designating at least... a daily snapshot*. Bhattacharjee, ¶240. This satisfies claim 5 if, due to the claim language “at least one of... and,” the claim can be met by *designating* any listed item. EX1001, 39:31-40.

If the claim requires that *one* snapshot collected at *one* designated daily time be treated as “*the*” daily snapshot, this would have been obvious. Designating specific snapshots as daily (or weekly/monthly/quarterly) snapshots was well-known. Bhattacharjee, ¶241 (citing corroborating EX1047, [0163]-[0165];

EX1023, [0023]; EX1048, [0060]; EX1049, [0164]; EX1050, [0053]; EX1051, [0025]). POSAs would have found it obvious to implement Sivasubramanian+Zha+Shekar's UI (*supra* §VI.D) to additionally accept input of the daily snapshot time, as this was a known UI field that beneficially let users choose the best daily snapshot time based on their own circumstances. Bhattacharjee, ¶241; RDS, 16-17, 187 (daily "backup window" during which "snapshot" is taken); EX1052, 11:14-20, Fig. 2. POSAs would have expected success because the UI implementation was within a POSA's skill. *Supra* §VI.A.2.

If the claim requires *all* the recited items, this would have been obvious; it was known to specify time different periods for capturing snapshots for different snapshot frequencies, and this would have provided the same benefits discussed in the previous paragraph for daily snapshot time designation and would have been within a POSA's skill. Bhattacharjee, ¶242 (citing corroborating EX1047, [0165]; EX1052, 10:20-22, 11:14-20, Fig. 2); *supra* §VI.A.2.

6. Claim 6

Sivasubramanian's Fig. 5 (annotated below) shows a field for a "Repository *name*" (502).

Web Browser

FILE EDIT VIEW FAVORITES TOOLS HELP

Address

GO

Control interface page

Repository name: Repository A

Task: ☒ Provision ☐ Create ☐ Delete ☐ Modify

Database engine: MySQL Version: 5

Storage capacity: 150 Units: GB

Partition scheme:

Security scheme: Version: 0

Access key: Version: 0

Save Cancel

POSAs understood that the “Repository name” in Figure 5 is a database *name*, as explained *supra* §VI.A.1.a.i. Thus, POSAs understood that the “repository name” in Sivasubramanian’s Fig. 5 GUI is a *database name for the database*. Bhattacharjee, ¶¶243-244.

POSAs understood that the Figure 5 GUI would display a field for entering a new “repository name,” *i.e. database name*, when the “Create” task is selected. *Supra* §VI.A.1.a.i. POSAs understood that the purpose of presenting a UI with a field for a value is to receive an input of that value; thus, POSAs understood that Sivasubramanian’s GUI receives *a name for the source database* via the GUI. Sivasubramanian, 5:59-62 (referencing “[w]eb service frontend features such as ... validating user *input* ...”), 12:55-57 (Fig. 5 UI lets user “enter information”);

11:49-55 (discussing control plane user “specify[ing]” a “name”). Bhattacharjee, ¶245.

Further, Sivasubramanian’s “rds-create-database” tool accepts the parameter “identifier” that is separate from the parameter “dbName.” Sivasubramanian, 13:1-5. POSAs would have likewise found it obvious to include instructions a field for receiving a similar “identifier” parameter as a database description in Sivasubramanian’s Figure 5 GUI; this would beneficially provide consistency with the command-line interface, and was within a POSA’s skill. Bhattacharjee, ¶246; *supra* §VI.A.2. POSAs understood that the “identifier” is an example of a database description. The ’157 patent’s only example of a “database “description,” is something arbitrary that a user can enter in a text input field 460B in Figure 4E, which is simply a text input field distinct from the “name” field 460A, as a *description for the source database.*” EX1001, 24:53-56; Bhattacharjee, ¶246.

7. Claim 7

Sivasubramanian *creates the source database* as part of the database-creation task launched from Figure 5’s “Create” GUI. *Supra* [1C], [1J]. In this scenario, *the provisioning comprises creating a new database as the source database*. Bhattacharjee, ¶248.

Further, Sivasubramanian’s “Provision” task in Figure 5 lets the user *provision a source database* that is an *existing database* identified using the “Repository name” drop-down menu. Sivasubramanian, Fig. 5; *supra* §VI.A.1.a.i.

The screenshot shows a web browser window titled "Web Browser" with a menu bar (FILE, EDIT, VIEW, FAVORITES, TOOLS, HELP) and an address bar. The main content area is titled "Control interface page".

Key elements and annotations:

- Repository name:** A text field containing "Repository A". A red circle highlights a small downward arrow icon to its right, with a red arrow pointing to it from the text "“Down arrow” control indicating a drop-down menu". A yellow box highlights the text "Repository name" with the number 504 next to it.
- Task:** A group of radio buttons labeled "Provision", "Create", "Delete", and "Modify". The "Provision" button is selected. A yellow box highlights the "Provision" button with the number 502 next to it.
- Database engine:** A text field containing "MySQL". A red circle highlights a small downward arrow icon to its right, with a red arrow pointing to it from the text "“Down arrow” control indicating a drop-down menu". A yellow box highlights the text "Database engine" with the number 506 next to it.
- Version:** A text field containing "5". A red circle highlights a small downward arrow icon to its right, with a red arrow pointing to it from the text "“Down arrow” control indicating a drop-down menu". A yellow box highlights the text "Version:" with the number 508 next to it.
- Storage capacity:** A text field containing "150". A red circle highlights a small downward arrow icon to its right, with a red arrow pointing to it from the text "“Down arrow” control indicating a drop-down menu".
- Units:** A text field containing "GB". A red circle highlights a small downward arrow icon to its right, with a red arrow pointing to it from the text "“Down arrow” control indicating a drop-down menu".
- Partition scheme:** A text field.
- Security scheme:** A text field.
- Access key:** A text field.
- Buttons:** "Save" and "Cancel" buttons at the bottom.

Sivasubramanian, Detail of Fig. 5 (annotated)

In Sivasubramanian+Zha+Shekar, Sivasubramanian’s database-provisioning UIs accept an SLA selection and protection schedule as Zha teaches. *Supra* §§VI.D, [1H]-[1I]. POSAs would have understood that setting a protection

schedule and SLA for an existing database *registers the existing database as the source database*, because this action sets up the existing database as the source database for capturing snapshots. *Supra* [1C], [1I]; Bhattacharjee, ¶¶249-250; EX1001, 4:29-34, 25:43-67, FIGS. 5A-5E (describing “registering” existing database by providing connection information for the database and then specifying an SLA and protection schedule for it).

The ’157 patent says “existing databases” that are registered “*may* have been created outside of the database system,” indicating this is not *required*. If claim 6 is found to require it, however, POSAs would have found it obvious to implement Sivasubramanian’s database-provisioning UIs to allow a control-plane user to *register* a previously-unknown *existing database*. POSAs understood this would beneficially enable performing Sivasubramanian’s workflows, including the “recovery and backup” workflow, on existing databases, replacing “tedious manual procedures” for those databases. Sivasubramanian, 1:46-52, 6:31-42; Bhattacharjee, ¶251. POSAs would have reasonably expected success with such a UI implementation. Sivasubramanian discloses that (1) the control plane accepts input of parameters used for database access, such as DNS address and port, and (2) the control plane can “provision and manage” *pre-existing* databases in a data plane controlled by a “provider” separate from the control plane’s “provider.” Sivasubramanian, 6:59-67, 9:37-44, 11:40-52; Bhattacharjee, ¶251. Implementing

Sivasubramanian's UIs to accept these parameters was within a POSA's skill.

Supra §VI.A.2.

The above-discussed "Create" and "Provision" tasks are *first* and *second* options *presented* in a *user interface*, *i.e.* the Figure 5 GUI, *upon receiving* the user's request to access the Figure 5 UI, which is a *user input to provision the source database* as discussed *supra* [1C]; the *processor executes* the recited *instructions* as explained *supra* [1B]. Bhattacharjee, ¶252.

8. Claim 8

Supra [1D] (engine type options presented on "Provision" and "Create" screens); Bhattacharjee, ¶253.

9. Claim 9

In Sivasubramanian+Zha+Shekar, SLAs are differentiated from each other by *levels* (*supra* §§VI.D.1-VI.D.2) and the *processor receives the selection* of an SLA via Sivasubramanian's Figure 5 GUI. *Supra* [1H] (§VI.E.1.i). POSAs would have found it obvious to implement the control plane to present the levels, to the as *selectable options* (*e.g.*, in a drop-down menu as disclosed in Sivasubramanian's Figure 5 (*e.g.* element 506)), and then receive the *selection of at least one level*, since this would beneficially result in SLA selection for the database.

Bhattacharjee, ¶254. Receiving such selection was within a POSA's skill. *Supra* §VI.A.2. Further, as previously discussed (*supra* [1I]),

Sivasubramanian+Zha+Shekar teaches *selection of a frequency of capturing the snapshot and the transactional log*.

POSAs would have found it obvious to customize Sivasubramanian’s GUI to present the selectable options claim 9 recites *upon receiving selection of the first profile and the second profile* recited in [1F]-[1G] because sequential UIs were obvious as discussed *supra* [1E]. Bhattacharjee, ¶255.

10. Claim 10

a. [10PRE]: “A method comprising:”

Sivasubramanian discloses a “computer-implemented *method* of controlling... a configuration aspect of a data environment using a control environment.” Sivasubramanian, claims 1, 4, 15; *see also* 2:17-18. Similar to claim 1, Sivasubramanian’s *method*, in view of the teachings of Zha and Shekar teachings as described *supra* §VI.D, *comprises* the remaining steps recited in claim 10. Bhattacharjee, ¶256.

b. [10A]-[10I]

Sivasubramanian+Zha+Shekar meets [10A]-[10I] for the same reasons as the corresponding limitations listed below and addressed above. Bhattacharjee, ¶¶257-265.

Claim 10 Limitation	Corresponding Limitation, <i>supra</i>
[10A]	[1C]

[10B]	[1D]
[10C]	[1E]
[10D]	[1F]
[10E]	[1G] ([1G] recites “ <i>creating</i> ” whereas [10E] recites “ <i>provisioning</i> ”; both “Create” and “Provision” in Sivasubramanian’s GUI perform <i>provisioning</i> ; e.g., <i>supra</i> §VI.A.1, [1C], [1G]).
[10F]	[1H]
[10G]	[1I]
[10H]	[1J]
[10I]	[1K]

11. Claims 11-14

These claims are met by Sivasubramanian+Zha+Shekar for the same reasons as the corresponding claims listed below and addressed above. Bhattacharjee, ¶¶266-269.

Claim	Corresponding Claim, <i>supra</i>
Claim 11	Claim 2
Claim 12	Claim 3
Claim 13	Claim 7
Claim 14	Claim 8

12. Claim 15

a. [15Pre]: “An apparatus comprising:”

As discussed *supra* [1Pre], Sivasubramanian discloses a “*system* for controlling... a configuration aspect of at least one data instance in [a] data environment.” Sivasubramanian, claims 20, 24; *see also* 2:17-18, 4:48-53. POSAs understood that Sivasubramanian’s system to include an *apparatus* for performing this control (the “control plane”), shown in FIG. 2. Bhattacharjee, ¶270.

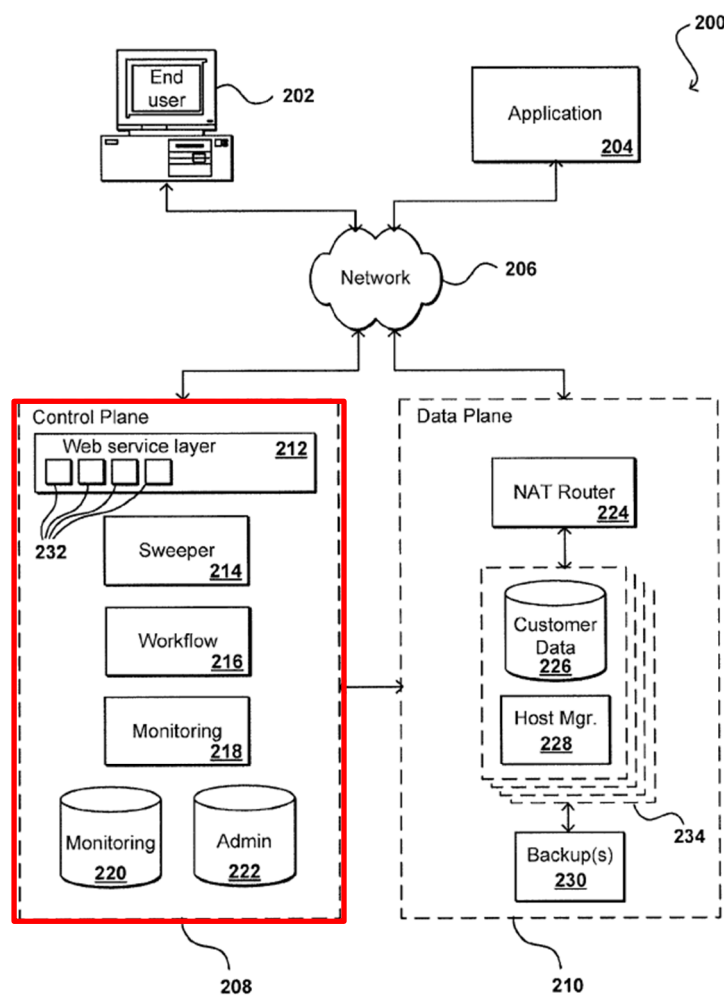


FIG. 2

Sivasubramanian, FIG. 2 (annotated)

b. [15A]-[15K]

These limitations are met by Sivasubramanian+Zha+Shekar for the same reasons as the corresponding limitations listed below and addressed above.

Bhattacharjee, ¶¶271-281.

Claim 15 Limitation	Corresponding Limitation, <i>supra</i>
[15A]	[1A]
[15B]	[1B]
[15C]	[1C]
[15D]	[1D]
[15E]	[1E]
[15F]	[1F]
[15G]	[1G]
[15H]	[1H]
[15I]	[1I]
[15J]	[1J]
[15K]	[1K]

13. Claims 16-19

These claims are met by Sivasubramanian+Zha+Shekar for the same reasons as the corresponding claims listed below and addressed above. Bhattacharjee, ¶¶282-285.

Claim	Corresponding Claim, <i>supra</i>
--------------	--

Claim 16	Claim 6
Claim 17	Claim 7
Claim 18	Claim 8
Claim 19	Claim 9

VII. GROUND 2: CLAIMS 1-19 ARE UNPATENTABLE OVER SIVASUBRAMANIAN+ZHA+SHEKAR+RDS

A. RDS (EX1013)

RDS is a user guide for “Amazon Relational Database Service (Amazon RDS),” which is “a web service that makes it easier to set up, operate, and scale a relational database in the cloud.” RDS, 1. RDS describes multiple GUIs that enable users to create and configure databases in a virtual computing system. *E.g.*, RDS, 13-15. Some of these GUIs include multiple screens (each a separate “*user interface*” in the ’157 patent’s terminology, *see infra* §VII.C.1) that are presented sequentially to the user to collect different portions of the information required to perform a task, such as creating a database. RDS, 12-17. Bhattacharjee, ¶286.

B. Sivasubramanian+Zha+Shekar+RDS Combination

POSAs would have found it obvious to further apply the user-interface teachings of RDS to the Sivasubramanian+Zha+Shekar combination described above, as described below.

As previously discussed throughout Ground 1 (*e.g.*, *supra* [1D]-[1I]), Sivasubramanian+Zha+Shekar teaches a GUI dashboard that receives selections of

various parameters from the user that are used to create and provision database instances in a virtual environment. In view of RDS, POSAs would have found it obvious to receive the selections of these parameters from user interface components presented on separate screens (or user interfaces, as discussed *infra* §VII.C.1-3) within the GUI dashboard. Bhattacharjee, ¶¶287-288.

POSAs would have been motivated to implement Sivasubramanian+Zha+Shekar’s GUI dashboard in this manner. The benefits of this type of sequential GUI—often referred to as a “wizard”—were well-known and the technique was in widespread use prior to the ’157 patent, especially in computing tasks having multiple phases like the database creation and provisioning process embodied by the Sivasubramanian+Zha+Shekar’s GUI dashboard. Bhattacharjee, ¶289 (citing corroborating EX1108, [0013]; EX1109, [0040]). By “guid[ing] the user through the task one step at a time,” a sequential GUI or “wizard,” like the GUI dashboard in Sivasubramanian+Zha+Shekar+RDS reduces the risk of user confusion by breaking a complex, multi-step task like database creation into multiple smaller subtasks, each presented on a separate screen of the GUI. EX1108, [0013]; EX1109, [0040]; Bhattacharjee, ¶289. POSAs would have been motivated by at least these benefits to implement Sivasubramanian+Zha+Shekar’s GUI dashboard according to the teachings of RDS.

The results of the combination would have been predictable because it was well-known to apply sequential GUI techniques like those described in RDS to “setup or configuration process[es]” such as the database creation and provisioning process embodied by the Sivasubramanian+Zha+Shekar’s GUI dashboard, and because such a UI implementation was within a POSA’s skill. Bhattacharjee, ¶290; Scott, [0040]; *supra* §VI.A.2.

C. Claim-by-Claim Analysis

1. Independent Claims 1, 10, and 15

With the exception of [1E], all claim 1 elements are met by Sivasubramanian+Zha+Shekar+RDS for the same reasons they are met by Sivasubramanian+Zha+Shekar as described in Ground 1. *Supra* §VI.E.1.

[1E] recites, *inter alia*, “*present[ing] a plurality of selectable options to be selected via at least one user interface of the GUI.*” If claim 1 is interpreted to require that *at least one user interface* to be at least one separate screen or view within the GUI, POSAs would have found it obvious for the GUI of Sivasubramanian+Zha+Shekar+RDS to present the *selectable options* of Limitations [1F]-[1I] as separate screens or “*user interfaces*” within the GUI, for example as part of a sequential GUI or wizard as described above. *Supra* §VII.B; Bhattacharjee, ¶¶291-292.

Elements [10D] and [15E] are analogous to [1E], and are thus met by Sivasubramanian+Zha+Shekar+RDS for the same reasons discussed above. All remaining claim 10 and 15 elements are met by Sivasubramanian+Zha+Shekar+RDS for the same reasons they are met by Sivasubramanian+Zha+Shekar as described in Ground 1. *Supra* §§VI.E.10, VI.E.12; Bhattacharjee, ¶293.

2. Claims 4-5, and 12-13

Claims 4-5, and 12-13 recite additional “*user interface[s]*” separate from the “*first user interface*” recited in their associated independent claims. *See* EX1001, claim 4 (“*second user interface*”), claim 5 (“*third user interface*”), claim 12 (“*second user interface*”), 13 (“*third user interface*”). If these claims are interpreted to require these recited user interfaces to be separate screens or views within the GUI, POSAs would have found it obvious for the GUI of Sivasubramanian+Zha+Shekar+RDS to present the selectable options of claims 4, 5, 12, and 13 as separate screens or “*user interfaces*” within the GUI, for example as part of a sequential GUI or wizard as described above. *Supra* §VII.B; Bhattacharjee, ¶294.

3. Claims 2-3, 6-9, 11, 14, and 16-19

The remaining dependent claims are met by Sivasubramanian+Zha+Shekar+RDS for the same reasons they are met by

Sivasubramanian+Zha+Shekar as described in Ground 1. *Supra* §§VI.E.2-3, VI.E.6-9, VI.E.11, VI.E.13, Bhattacharjee, ¶295.

VIII. NO BASIS EXISTS FOR DISCRETIONARY DENIAL

A. 314(a)

The *Fintiv* factors weigh against denial.

1. Stay Potential

Although no stay **in favor of IPR** has yet been requested, the district court recently granted Petition’s motion to stay in favor of arbitration (EX1122); Factor 1 weighs against denial based on this “specific evidence” of the court’s stay inclinations. *Sand Revolution II v. Continental Intermodal Group-Trucking*, IPR2019-01393, Paper 24, 7 (June 16, 2020) (informative).

2. Trial Timing

The district-court proceeding was recently stayed. EX1122. The Northern District of California’s median time to trial is 34.5 months (EX1123, at 66), and no post-stay schedule has been set. Factor 2 therefore weighs against denial. *BMW of North America, LLC v. Michigan Motor Techs., LLC*, IPR2023-01224, Paper 15, 11 (Feb. 15, 2024).

3. Litigation Investment

This weighs against denial. Petitioner filed this petition before any substantive orders in the litigation or significant investment. *Apple v. Seven Networks*, IPR2020-00156, Paper 10, 11-12 (June 15, 2020) (quoting *Fintiv*, 11).

4. Issue overlap

This weighs against denial. Petitioner challenges all claims; PO has not served contentions. The Board should institute to enhance patent system integrity and meet Petitioner's interest in resolving invalidity challenges on all claims. *Seven*, 18.

5. Litigation Defendants

This is neutral; Petitioner is a litigation defendant, which is not unusual. *Sand Revolution II*, 12-13.

6. Other Circumstances

The Petition's "strong showing on the merits" weighs against denial. *Seven*, 20.

B. 325(d)

Considering the two-part framework discussed in *Advanced Bionics, LLC v. Med-El Elektromedizinische Gerate GMBH*, IPR2019-01469, Paper 6 (Feb. 13, 2020) (precedential), the Board should not exercise its §325(d) discretion to deny institution.

1. Part One

As the Director recently noted in a notice of proposed rulemaking, "application of section 325(d) should be limited to situations in which the prior art or arguments were *meaningfully* addressed by the Office"—meaning that the Office "*articulated* its consideration of the art or arguments *in the record*." 89

Fed. Reg. 28693, 28700 (Apr. 19, 2024). The record here lacks any such articulation. *See* EX1002. Although Zha and a descendant of Sivasubramanian are listed on an IDS signed by the examiner (EX1002, 182, 289), the examiner did not articulate any consideration of them; nor is there any evidence that the Examiner considered their specific disclosures cited in the Grounds herein for the allegedly undisclosed limitations, nor that the examiner considered Sivasubramanian and Zha in combination. That weighs against discretionary denial. PTAB Rules for 325(d) Considerations, 89 Fed. Reg. 28693, 28705 (2024) (to be codified at 37 C.F.R. §42.108(f)) (proposed Apr. 19, 2024) (discretionary denial under §325(d) applies only if prior art was “meaningfully addressed”). Further, under *Becton, Dickinson* Factor (d), the Examiner never assessed patentability using the ***combinations*** in the Grounds (or any combination at all).

2. Part Two

If the Board reaches *Advanced Bionics* step two because Zha and a descendant of Sivasubramanian were of record, “specific teachings of [both references] impact[ing] patentability of the Challenged Claims” were “misapprehend[ed] or overlook[ed].” *Cellco P’ship v. Huawei Device Co.*, IPR2020-01117, Paper 10 at 12 (Feb. 3, 2021). The “fact that” neither reference formed a “basis of rejection weighs strongly against exercising [the Board’s] discretion to deny institution under 35 U.S.C. § 325(d).” *Id.* at 13. Furthermore,

the three “*Becton, Dickson*” (“*BD*”) factors applicable to Part Two (*Advanced Bionics*, 10) weigh against denial.

Factor (c) weighs against denial because the examiner did not use any reference from the Grounds in a rejection. *Group III v. Targus*, IPR2021-00371, Paper 21, 33 (July 9, 2021); 89 Fed. Reg. 28693, 28705 (325(d) discretionary denial only applies if art was “meaningfully addressed”).

The examiner did not identify or explain any claim limitation that either Sivasubramanian or Zha, or their combination, failed to disclose. Allowing the claims was error material to patentability because—as the Grounds explain—Sivasubramanian and Zha in combination with each other and other references render all claims obvious. Factor (e) thus weighs against denial.

The prosecution did not include the expert testimony and evidence of a POSA’s background knowledge presented in this Petition regarding why the claims are obvious. Thus, Factor (F) weighs against denial.

IX. CONCLUSION

The Board should institute review and cancel claims 1-19.

Dated: March 20, 2025

Respectfully submitted,
Tessell, Inc.

/Anant Saraswat/
Anant K. Saraswat, Reg. No. 76,050
WOLF, GREENFIELD & SACKS, P.C.

X. CLAIM LISTING

The following claim listing assigns element labels (e.g., [1A1], [1A2], etc.) to certain claims for clarity.

Claim 1
[1PRE] A system comprising:
[1A] a memory storing computer-readable instructions in a virtual computing system; and
[1B] a processor executing the computer-readable instructions to:
[1C] present a dashboard to receive a user input in a graphical user interface (GUI) to provision a source database;
[1D] upon receiving the user input, present options in a first user interface of the GUI to receive a first selection of a database engine type;
[1E] upon receiving the first selection of the database engine type, present a plurality of selectable options to be selected via at least one user interface of the GUI, the plurality of selectable options comprising at least a plurality of:
[1F] a second selection of a first profile for creating a source database virtual machine;
[1G] a third selection of a second profile for creating the source database;
[1H] a fourth selection of a Service Level Agreement (“SLA”); and
[1I] a fifth selection of a frequency of capturing a snapshot and transactional log;
[1J] upon receiving the selections of the database engine type and the at least plurality of the first profile, the second profile, the SLA, and the frequency of capturing the snapshot and transactional log, create the source database and associate the created source database with the source database virtual machine; and
[1K] display a status of the created source database on the dashboard.
Claim 2
The system of claim 1, wherein the dashboard comprises a dynamically changing home page comprising a plurality of cells, wherein a first cell of the plurality of cells displays a list of all source databases in the virtual computing system.
Claim 3

The system of claim 1, wherein to receive the selection of the first profile, the second user interface comprises fields for receiving selection of at least one of a software profile, a compute profile, and a network profile.
Claim 4
The system of claim 1, wherein to receive the selection of the first profile, the second user interface comprises fields for receiving selection of a name and description for the source database virtual machine.
Claim 5
The system of claim 1, wherein the third user interface further comprises fields for receiving selection of a duration for designating at least one of a daily snapshot, a weekly snapshot, a monthly snapshot, and a quarterly snapshot.
Claim 6
The system of claim 1, wherein the plurality of selectable options further comprise a sixth selection of a name and description as the second profile for the source database.
Claim 7
The system of claim 1, wherein upon receiving the user input to provision the source database, the processor executes computer-readable instructions to present a user interface providing a first option to create a new source database and a second option to register an existing database as the new source database.
Claim 8
The system of claim 7, wherein upon receiving selection of the first option, the system presents the options for the database engine type to be selected.
Claim 9
[9PRE] The system of claim 1, wherein the plurality of selectable options further comprise
[9A] a sixth selection of at least one level of the SLA and the frequency of capturing the snapshot and the transactional log upon receiving selection of the first profile and the second profile.
Claim 10
[10PRE] A method comprising:
[10A] presenting a dashboard to receive an user input in a graphical user interface (GUI) to provision a source database;

[10B] upon receiving the user input, presenting options in a first user interface of the GUI to receive a first selection of a database engine type;
[10C] upon receiving the first selection of the database engine type, presenting a plurality of selectable options to be selected via at least one user interface of the GUI, the plurality of selectable options comprising at least a plurality of:
[10D] a second selection of a first profile for creating a source database virtual machine;
[10E] a third selection of a second profile for provisioning the source database;
[10F] a fourth selection of a Service Level Agreement (“SLA”);
[10G] a fifth selection of a frequency of capturing a snapshot and transactional log;
[10H] upon receiving the selections of the database engine type and the at least plurality of the first profile, the second profile, the SLA, and the frequency of capturing the snapshot and transactional log, creating the source database and associate the created source database with the source database virtual machine; and
[10I] displaying a status of the created source database on the dashboard.
Claim 11
The method of claim 10, further comprising dynamically changing a home page comprising a plurality of cells, wherein a first cell of the plurality of cells displays a list of all source databases in the virtual computing system.
Claim 12
The method of claim 10, wherein the second user interface comprises fields for receiving selection of at least one of a software profile, a compute profile, and a network profile.
Claim 13
The method of claim 10, wherein upon receiving the user input, the method further comprises, presenting at the dashboard a user interface providing a first option to create a new source database and a second option to register an existing database as the new source database.
Claim 14
The method of claim 13, further comprising upon receiving selection of the first option, receiving the selection of the database engine type.

Claim 15
[15PRE] An apparatus comprising:
[15A] a memory comprising computer-readable instructions stored thereon; and
[15B] a processor that executes the computer-readable instructions to:
[15C] present a dashboard to receive an user input in a graphical user interface (GUI) to provision a source database;
[15D] upon receiving the user input, present options in a first user interface of the GUI to receive a first selection of a database engine type;
[15E] upon receiving the first selection of the database engine type, present a plurality of selectable options to be selected via at least one user interface of the GUI, the plurality of selectable options comprising at least a plurality of:
[15F] a second selection of a first profile for creating a source database virtual machine;
[15G] a third selection of a second profile for creating the source database;
[15H] a fourth selection of a Service Level Agreement (“SLA”);
[15I] a fifth selection of a frequency of capturing a snapshot and transactional log;
[15J] upon receiving the selections of the database engine type and the at least plurality of the first profile, the second profile, the SLA, and the frequency of capturing the snapshot and transactional log, create the source database and associate the created source database with the source database virtual machine; and
[15K] display a status of the created source database on the dashboard.
Claim 16
The apparatus of claim 15, wherein the plurality of selectable options further comprise a sixth selection of a name and description as the second profile for the source database.
Claim 17
The apparatus of claim 15, wherein upon receiving the user input to provision the source database, the processor executes computer-readable instructions to present a user interface providing a first option to create a new source database and a second option to register an existing database as the new source database.
Claim 18

The apparatus of claim 17, wherein upon receiving selection of the first option, the system presents the options for the database engine type to be selected.

Claim 19

The apparatus of claim 15, wherein the plurality of selectable options further comprise a sixth selection of at least one level of the SLA and the frequency of capturing the snapshot and the transactional log upon receiving selection of the first profile and the second profile.

CERTIFICATE OF SERVICE UNDER 37 C.F.R. § 42.6 (E)(4)

I certify that on March 20, 2025, a copy of the foregoing document, including any exhibits or appendices filed therewith, is being served via Priority Express Mail at the following correspondence address of record for the patent:

FOLEY & LARDNER LLP
3000 K STREET N.W.
SUITE 600
WASHINGTON, DC 20007-5109

Date: March 20, 2025

/MacAulay Rush/
MacAulay Rush
Paralegal
WOLF, GREENFIELD & SACKS, P.C.

CERTIFICATE OF WORD COUNT

Pursuant to 37 C.F.R. § 42.24, the undersigned certifies that the foregoing Petition for *Inter Partes* Review contains 13,999 words excluding a table of contents, a table of authorities, Mandatory Notices under § 42.8, a certificate of service or word count, or appendix of exhibits or claim listing. Petitioner has relied on the word count feature of the word processing system used to create this paper in making this certification.

Date: March 20, 2025

/MacAulay Rush/
MacAulay Rush
Paralegal
WOLF, GREENFIELD & SACKS, P.C.