

UNITED STATES PATENT AND TRADEMARK OFFICE

---

BEFORE THE PATENT TRIAL AND APPEAL BOARD

---

**DECLARATION OF BENJAMIN B. BEDERSON, PH.D.  
IN SUPPORT OF PETITION FOR INTER PARTES REVIEW OF  
U.S. PATENT 8,209,317**

## Table of Contents

|       |                                                                                    |    |
|-------|------------------------------------------------------------------------------------|----|
| I.    | Introduction and Background .....                                                  | 1  |
| II.   | Qualifications.....                                                                | 2  |
| III.  | Legal Standards .....                                                              | 8  |
|       | A. Overview .....                                                                  | 8  |
|       | B. Claim Construction.....                                                         | 9  |
|       | C. Person of Ordinary Skill in the Art.....                                        | 10 |
|       | D. Anticipation .....                                                              | 10 |
|       | E. Obviousness.....                                                                | 11 |
|       | F. Secondary Considerations of Nonobviousness.....                                 | 13 |
| IV.   | Level of Ordinary Skill in the Art .....                                           | 15 |
| V.    | Overview of the '317 Patent.....                                                   | 16 |
|       | A. Overview of the Technology .....                                                | 16 |
|       | B. Overview of the Prosecution History .....                                       | 18 |
| VI.   | Interpretation of Claim Terms .....                                                | 22 |
| VII.  | Overview of the Prior Art.....                                                     | 23 |
|       | A. Background and State of the Art for the '317 Patent .....                       | 23 |
|       | B. Overview of Sareen .....                                                        | 25 |
|       | C. Overview of Anick .....                                                         | 28 |
|       | D. Overview of Aravamudan .....                                                    | 30 |
|       | E. Overview of Kamvar .....                                                        | 32 |
| VIII. | Ground 1: Claims 1-11 are obvious in view of Sareen in combination with anick..... | 35 |
|       | A. Claim 1 .....                                                                   | 35 |
|       | 1. [1p] Preamble .....                                                             | 35 |
|       | 2. [1a] Database .....                                                             | 36 |
|       | 3. [1b] Query Reconstruction Server.....                                           | 39 |
|       | 4. [1c] Full Query Better Representative of Entire Query.....                      | 49 |
|       | B. Claim 2 .....                                                                   | 50 |
|       | C. Claim 3 .....                                                                   | 51 |
|       | D. Claim 4 .....                                                                   | 53 |
|       | E. Claim 5 .....                                                                   | 56 |
|       | F. Claim 6 .....                                                                   | 58 |
|       | G. Claim 7 .....                                                                   | 59 |

|     |                                                                                                             |    |
|-----|-------------------------------------------------------------------------------------------------------------|----|
| H.  | Claim 8 .....                                                                                               | 60 |
| 1.  | [8p] Preamble .....                                                                                         | 60 |
| 2.  | [8a] Storing in Memory .....                                                                                | 61 |
| 3.  | [8b] Retrieving Data from the Memory.....                                                                   | 61 |
| 4.  | [8c] Constructing Indexes.....                                                                              | 63 |
| 5.  | 8[d] Receiving a Partial Query .....                                                                        | 63 |
| 6.  | [8e] Determining a Full Query .....                                                                         | 64 |
| I.  | Claim 9 .....                                                                                               | 65 |
| J.  | Claim 10 .....                                                                                              | 66 |
| K.  | Claim 11 .....                                                                                              | 68 |
| IX. | Ground 2: Claims 12-14 are obvious in view of Sareen in combination with Anick and aravamudan.....          | 71 |
| A.  | Claim 12 .....                                                                                              | 71 |
| B.  | Claim 13 .....                                                                                              | 73 |
| C.  | Claim 14 .....                                                                                              | 74 |
| X.  | Ground 3: claims 15-18 are obvious in view of Sareen in combination with kamvar, Aravamudan and Anick. .... | 75 |
| A.  | Claim 15 .....                                                                                              | 75 |
| 1.  | [15p] Preamble .....                                                                                        | 75 |
| 2.  | [15a] Storing in Memory .....                                                                               | 75 |
| 3.  | [15b] Receiving a Partial Query .....                                                                       | 78 |
| 4.  | [15c] Interpreting Each Word of the Partial Query as a Partial Word .....                                   | 78 |
| 5.  | [15d] Identifying Matching Queries .....                                                                    | 79 |
| 6.  | [15e] Selecting a Full Query.....                                                                           | 79 |
| B.  | Claim 16 .....                                                                                              | 80 |
| C.  | Claim 17 .....                                                                                              | 80 |
| D.  | Claim 18 .....                                                                                              | 81 |
| XI. | Conclusion.....                                                                                             | 82 |

## **I. INTRODUCTION AND BACKGROUND**

I, Benjamin B. Bederson, hereby declare as follows:

1. I submit this declaration in support of American Airlines' and Hilton's ("Petitioners'") Petition for Inter Partes Review of U.S. Patent 8,209,317 ("the '317 patent").

2. I understand that my curriculum vitae has been submitted as Exhibit 1004.

3. In preparing this declaration, in addition to my knowledge and experience, I have reviewed and am familiar with the following references:

- The '317 patent (EX1001)
- File History of the '317 patent (EX1002)
- U.S. Patent Application Publication 2007/0203890 to Sareen et al. ("Sareen," EX1007)
- U.S. Patent 6,947,930 to Anick et al. ("Anick," EX1008)
- U.S. Patent Application Publication 2007/0005563 to Aravamudan *et al.* ("Aravamudan," EX1009)
- U.S. Patent Application Publication No. 2005/0283468 to Kamvar *et al.* ("Kamvar," EX1010)

I have also considered all other materials cited herein.

## II. QUALIFICATIONS

4. Since 1998, I have been a Professor of Computer Science at the University of Maryland (“UMD”), where I have joint appointments at the Institute for Advanced Computer Studies and the College of Information Studies (Maryland’s “iSchool”) and am currently Professor Emeritus. I was also Associate Provost of Learning Initiatives and Executive Director of UMD’s Teaching and Learning Transformation Center from 2014 to 2018. I am a member and previous director of the Human-Computer Interaction Lab (“HCIL”), the oldest Human-Computer Interaction research group in the country, and one of the best-known. Last year, I co-founded the J.S. Bryant School, a therapeutic high school to launch in 2025. From 2006 to 2014, I was also co-founder and Chief Scientist of Zumobi, Inc., a Seattle-based startup that is a publisher of content applications and advertising platforms for smartphones. I am a co-founder and co-director of the International Children’s Digital Library (“ICDL”), a website launched in 2002 that provides the world’s largest collection of freely available online children’s books from around the world and has an interface aimed to make it easy for children and adults to search and read children’s books online. From 2014-2023, I was also co-founder and prior Chief Technology Officer of Hazel Analytics, a data analytics company whose product sends alerts in warranted circumstances. In addition, for more than 25 years I have consulted in the area of user interfaces for numerous companies, including EPAM,

Logitech, Microsoft, the Palo Alto Research Center, Sony, Lockheed Martin, and the NASA Goddard Space Flight Center.

5. I received a Bachelor of Science degree in Computer Science with a minor in Electrical Engineering from Rensselaer Polytechnic Institute (“RPI”) in 1986. I received a Master of Science degree and a Ph.D. in Computer Science from New York University (“NYU”) in 1989 and 1992, respectively.

6. For more than 30 years, I have studied, designed, and worked in the field of computer science and human-computer interaction. My experience includes 30 years of teaching and research, with research interests in human-computer interaction and the software and technology underlying today’s interactive computing systems. This includes the design and implementation of user interfaces and software applications, including web browsers and search engines as described below.

7. At UMD, my research is in the area of Human-Computer Interaction (“HCI”), a field that relates to the development and understanding of computing systems to serve users’ needs. Researchers in this field are focused on making universally usable, useful, efficient, and appealing systems to support people in a wide range of activities. My approach is to balance the development of innovative technology that serves people’s practical needs. Following this approach, I have built the following systems: Pad++ (1994 software for zoomable user interfaces), KidPad

(1997 software to help children collaboratively write stories), PadPrints (1998 software to graphically show interactive web browsing histories), PhotoMesa (2001 software to help end users browse personal photos), Pocket PhotoMesa (2004 software for browsing personal photos on mobile devices), DateLens (2002 software to help end users use their mobile devices to efficiently access calendar information), LaunchTile (2005 “home screen” software for mobile devices to allow users to navigate apps in a zoomable environment), SpaceTree (2001 software to help end users efficiently browse very large hierarchies), ICDL (2002 International Children's Digital Library), and Story Kit (2009 iPhone application to help children create stories).

8. Pad++, which I wrote in 1994, included a number of techniques to support high performance interactive visualization.<sup>1</sup> Pad++ included a custom web browser that I built from scratch and distributed as open source software at the Pad++ website. The web browser was described in numerous publications, including “Pad++: A Zoomable Graphical Interface for Exploring Alternate Interface Physics.”<sup>2</sup> For the web browser, I implemented from scratch a rendering engine that

---

<sup>1</sup> Bederson, B.B. & Meyer, J. (1998). Implementing a Zooming User Interface: Experience Building Pad++. *Software: Practice and Experience*, 28(10), 1101-1135.

<sup>2</sup> Bederson, B. B. & Hollan, J. D. (1994). “Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics,” *ACM Conference on User Interface Software and Technology*, 17-26.

supported the version of HTML, available at the time (HTML, 1.0) as well as an embedded scripting language.

9. By 1998, my team had combined Pad++ with Netscape Navigator (a standard web browser of the time) to offer a graphical web history, visually showing what pages a user had visited with an application we called “PadPrints.”<sup>3</sup> We did this by implementing a web “proxy,” which was a standard way of modifying the built-in behavior of web browsers at the time. Some proxies work by intercepting all web traffic coming from and going to a web browser. Once intercepted, the proxy can do whatever it wants with the traffic, such as caching it for better performance, simply allowing the traffic through and monitoring it, modifying the data itself, or holding that connection and opening new connections. PadPrints’ proxy worked by monitoring web traffic and letting PadPrints know what the user was doing so PadPrints could display the web history.

10. I have since 2002 been the technical director responsible for the design and implementation of the International Children’s Digital Library that can be found at [www.childrenslibrary.org](http://www.childrenslibrary.org) (originally at [www.icdlbooks.org](http://www.icdlbooks.org)). In particular, I have been closely involved in designing the user interface and software architecture for

---

<sup>3</sup> Ron R. Hightower, Laura T. Ring, Jonathan I. Helfman, Benjamin B. Bederson & James D. Hollan. 1998. PadPrints: Graphical Multiscale Web Histories. In Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology (UIST ’98). Association for Computing Machinery, New York, NY, USA, 121-122. <https://doi.org/10.1145/288392.288582>

the website since its inception in 2002. The ICDL is the world's largest collection of freely available children's books from around the world. From the very beginning, it included a search engine that supported searching for books in the library based on book metadata such as title, genre, and color of the corresponding book cover. I was part of the team that designed this search engine functionality.

11. Since the mid-1990s, I have been responsible for the design and implementation of numerous other websites in addition to the ICDL. For example, I designed and built my own professional website when I was an Assistant Professor of Computer Science at the University of New Mexico in 1995, and I have continued to design, write code for, and update both that site (which I moved to the University of Maryland in 1998, currently at <http://www.cs.umd.edu/~bederson/>) and numerous project websites, such as Pad++ (located at <http://www.cs.umd.edu/hcil/pad++/>). I received the Janet Fabri Memorial Award for Outstanding Doctoral Dissertation for my Ph.D work in robotics and computer vision. I have combined my hardware and software skills throughout my career in human-computer interaction ("HCI") research, building various interactive electrical and mechanical systems that couple with software to provide an innovative user experience.

12. My work has been published extensively in more than 160 technical publications, and I have given about 100 invited talks, including 9 keynote lectures. I have won a number of awards, including the Brian Shackel Award for Outstanding

Contribution with International Impact in the Field of HCI in 2007 and the Social Impact Award from Association for Computing Machinery's ("ACM") Special Interest Group on Computer Human Interaction ("SIGCHI") in 2010. ACM is the primary international professional community of computer scientists, and SIGCHI is the primary international professional HCI community. I have been honored by both professional organizations. I am an "ACM Distinguished Scientist," which "recognizes those ACM members with at least 15 years of professional experience and 5 years of continuous Professional Membership who have achieved significant accomplishments or have made a significant impact on the computing field." I am a member of the "CHI Academy," which is described as follows: "The CHI Academy is an honorary group of individuals who have made substantial contributions to the field of human-computer interaction. These are the principal leaders of the field, whose efforts have shaped the disciplines and/or industry, and led the research and/or innovation in human-computer interaction." The criteria for election to the CHI Academy are: (1) cumulative contributions to the field; (2) impact on the field through development of new research directions and/or innovations; and (3) influence on the work of others. I have received two "Test of Time" awards from IEEE InfoVis for my 2001 and 2002 work on visualizing hierarchies, including those containing digital photos.

13. I have appeared on radio shows numerous times to discuss issues relating to user interface design and people's use and frustration with common technologies, websites, and mobile devices. My work has been discussed, and I have been quoted, by mainstream media around the world over 120 times, including by *The New York Times*, *The Wall Street Journal*, *The Washington Post*, *Newsweek*, the *Seattle Post-Intelligencer*, *The Independent*, *Le Monde*, NPR's *All Things Considered*, *New Scientist Magazine*, and MIT's *Technology Review*.

14. I have designed, programmed, and publicly deployed dozens of user-facing software products that have cumulatively had millions of users. I am the co-inventor of 14 U.S. patents and 20 U.S. patent applications, which are generally directed to user interfaces and experience.

### **III. LEGAL STANDARDS**

#### **A. Overview**

15. I am not an attorney or a legal expert, and I do not opine on matters of law. All the legal standards listed in this section, and which I applied in reaching the conclusions expressed throughout this declaration, were provided to me by counsel. In preparing this declaration, I have endeavored to faithfully apply these legal standards. I do not intend to offer opinions on the law. Below, I describe the law that I applied in analyzing the claims of the '317 patent.

16. I also understand that an unpatentability analysis is conducted in two steps. First, the claims of the patent are construed to determine their meaning and scope. Second, after the claims are construed, the content of the prior art is compared to the construed claims.

17. I understand that a claimed invention is only patentable when it is new, useful, and non-obvious in light of the prior art. That is, the invention, as defined by the claims of the patent, must not be anticipated or rendered obvious by the prior art.

**B. Claim Construction**

18. I have been informed by counsel that the first step in determining validity is to properly construe the claims. I have further been informed by counsel that, in this proceeding, the Board must determine the scope of the claims by using the same claim construction standard that United States District Courts use. Under this standard, words of a claim are given their “ordinary” and “customary” meaning as would have been understood by a person of skill in the art (“POSITA”) at the time of the invention. I have been informed that such a person looks to the claims, specification, and file history in understanding the meaning of the claim terms. I understand that, in the specification, a patentee may define his own terms, give a claim term a different meaning than it would otherwise possess, or disclaim or disavow some claim scope.

19. I understand that in addition to the claims, specification, and prosecution history, other evidence may be considered to ascertain the meaning of claim terms, including textbooks, encyclopedias, articles, and dictionaries. I have been informed that this other evidence is often less significant and less reliable than the claims, specification, and prosecution history.

**C. Person of Ordinary Skill in the Art**

20. I have been informed that the patentability of a claimed invention must be analyzed from the perspective of “one of ordinary skill in the art” in the same field as the patent-in-suit at the time of the invention. I have also been informed that several factors are considered in assessing the level of ordinary skill in the art, including (1) the types of problems encountered in the art; (2) the prior art solutions to those problems; (3) the rapidity with which innovations are made; (4) the sophistication of the technology; and (5) the education level of active workers in the field. When I discuss in this declaration what a person of ordinary skill in the art (or synonymously a skilled artisan) would have known or understood, I am referring to at the time of the invention, which I understand to be August 10, 2006.

**D. Anticipation**

21. I understand that a patent claim is invalid as anticipated if a single piece of prior art teaches every element of the claims, viewed from the perspective of a person of ordinary skill in the art. I also understand that an anticipatory reference

does not have to recite word for word what is in the anticipated claims. Anticipation can also occur when a claimed limitation is inherent in the relevant reference. I have been advised that if the prior art necessarily functions in accordance with, or includes, a claimed limitation, it can anticipate even though the limitation is not expressly disclosed.

**E. Obviousness**

22. I understand that a patent claim is invalid as obvious if the claimed invention would have been obvious to a POSITA at the time the claimed invention was made. In this context, “obvious” means that even if all of the elements of a claimed invention cannot be found in a single prior art reference that would anticipate the invention, a POSITA who knows about all the prior art would have come up with the invention. I understand that in an obviousness determination, a POSITA is presumed to have knowledge of all material prior art. I understand that whether a claim is obvious is based upon a determination of several factual issues.

23. In considering obviousness, I understand that one must determine the scope and content of the prior art. I understand that, in order to be considered “prior art” to a patent under consideration, a reference must be reasonably related to the claimed invention of that patent. A reference is reasonably related if it originates from the same field as the claimed invention or from another field to which a POSITA would look to solve a known problem.

24. I understand that an assessment of obviousness requires a determination as to what differences, if any, existed between the claimed invention and the prior art.

25. I understand that a patent claim composed of several elements is not proved obvious merely by demonstrating that each of its elements was independently known in the prior art. In evaluating whether a claim would have been obvious, one may consider whether one or more reasons existed that would have prompted a POSITA to combine the elements or concepts from the prior art in the same way as in the claimed invention. There is no single way to define the line between true inventiveness on the one hand (which is patentable) and the application of common sense and ordinary skill to solve a problem on the other hand (which is not patentable). For example, market forces or other design incentives, rather than true inventiveness, may inspire a given change.

26. I understand that obviousness requires a showing that a skilled artisan would have been motivated to combine the teachings of the prior art references to achieve the claimed invention, and that a skilled artisan would have had a reasonable expectation of success in doing so. I understand that one may consider whether (1) the change was merely the predictable result of using prior art elements according to their known functions or the result of true inventiveness; (2) there was some teaching or suggestion in the prior art to make the modification or combination

of elements claimed in the patent; (3) the innovation applied a known technique that had been used to improve a similar device or method in a similar way; (4) the claimed invention would have been obvious to try, meaning that the claimed innovation was one of a relatively small number of possible approaches to the problem that a POSITA could have implemented with a reasonable expectation of success; (5) the invention merely substituted one known element for another known element to obtain predictable results; (6) the invention merely applied a known technique to a known device, method, or product to yield predictable results; or (7) known work in the field prompted variations in use of the same inventions in the same or different fields due to market forces or design incentives that would have been predictable to a POSITA.

27. I understand that one must be careful not to determine obviousness using the benefit of hindsight, as many true inventions might seem obvious after the fact. I understand that the determination should be based on the position of a POSITA at the time the claimed invention was made and should not account for what is known today or what is learned from the teachings of the patent under consideration.

**F. Secondary Considerations of Nonobviousness**

28. I understand that certain secondary considerations may be considered in evaluating obviousness in order to prevent hindsight bias. These secondary

considerations include commercial success of products that practice the patent, any long-felt need for the patented technology, failure by others to solve the problem addressed by the patent, initial skepticism by others in the industry, industry recognition and praise of the patented products, and efforts by others to copy the patented technology.

29. I have been informed that these secondary considerations of nonobviousness must be balanced against the strength of the prima facie case of obviousness. Where the invention represents no more than the predictable use of prior art elements according to their established functions, secondary considerations are inadequate to establish nonobviousness.

30. I have also been informed that the patentee must establish a nexus between any secondary consideration factors and the claimed invention. Where a secondary consideration factor results from something other than what is both claimed and novel in the patent, there is no nexus to the merits of the claimed invention.

31. I have been informed that, for example, the need for a nexus as it applies to the evidence-of-commercial-success factor requires that the patentee show that the sales of a product that allegedly practices the claimed invention are or were a direct result of the unique characteristics of the claimed invention, as opposed to other economic and commercial factors unrelated to the patented invention. I

understand that this means that if the product's commercial success is due to an element that existed in the prior art or that is unclaimed by the patent, then no nexus exists.

32. I have been further informed that, for example, the nexus requirement as it applies to the long-felt-need factor must likewise be supported by evidence. Where differences between the prior art and the claimed invention are minimal, it cannot be said that any long-felt need was satisfied by the patented invention or that the patent solved any technological problems left unaddressed by others.

33. I have been informed that evidence of industry praise must also show a nexus with the claimed invention, and that self-serving statements by the patentee regarding the merits of the claimed invention are generally given no weight.

34. I am not presently aware of any evidence of secondary considerations suggesting the claimed systems and methods are not obvious, and I reserve my right to address any such evidence if it is identified in the future.

#### **IV. LEVEL OF ORDINARY SKILL IN THE ART**

35. In my opinion, a POSITA pertinent to the '317 patent as of its earliest priority date would have had at least a four-year degree in computer science, computer engineering, or a closely related field and one to two years of experience in researching, designing, or developing computerized search systems such as search engines. Additional formal education could substitute for professional experience,

and significant professional experience could substitute for formal education. I meet at least this definition of a POSITA now, and I did so as of the priority date of the '317 patent.

36. I am also familiar with the knowledge of a POSITA as of the priority date of the '317 patent. I can opine on how the person of ordinary skill in the art would have understood the disclosure and claims of the '317 patent, the disclosures of the prior art, motivations to combine the prior art,<sup>4</sup> and the combinations that would have been obvious to a POSITA.

## **V. OVERVIEW OF THE '317 PATENT**

### **A. Overview of the Technology**

37. The '317 Patent is entitled “Method and Apparatus for Reconstructing a Search Query” and lists Deepa Joshi as the sole named inventor and Yahoo! Inc. as the Assignee. The '317 Patent discloses computer database systems and methods for a computer database system that reconstruct partial queries submitted by users at remote client systems over a network. The abstract of the '317 Patent states that “[e]xisting interfaces for search engines may be rigid and require users to submit full queries to perform searche[s]. The methods and systems described herein may solve these problems by allowing a flexible way for users to submit a partial query and

---

<sup>4</sup> As used herein, the phrases “found it obvious” and “motivated to combine” are used as shorthand to refer to a POSITA’s motivation to combine with a reasonable expectation of success.

reconstruct a full query based on the partial query. A search may then be performed using the reconstructed query.”

38. The methods and systems of the ’317 Patent are illustrated by Figure 2. Figure 2 provides an illustration of an exemplary process for reconstructing partial queries submitted by users at remote client systems over a network. Reconstructed full queries are determined based on the received partial query and stored information about queries previously submitted by users, and search results for the reconstructed full queries may later be presented to the users alongside alternative reconstructed full queries.

39. At 210, a user submits a partial query, which is received by a server. ’317 Patent, 3:10-59. As explained by the Applicant during the underlying prosecution history, the user must commit a deliberate and conscious act of submission of the partial query (*see* Section V.B. below). The specification of the ’317 Patent states that the term “‘partial query’ means any abbreviated or incomplete search query such that the submitted query is not fully representative of the entire search query desired by the user.” *Id.* at 3:12-15.

40. At 220, a full query is then reconstructed from the partial query, for instance “auto insurance” or “American Idol<sup>TM</sup>” could be reconstructed from the partial query “a i”. *Id.* at 3:10-59. The ’317 Patent states that the determination of the full query may be based on previously submitted queries and/or indexes of

previously submitted queries that were received by a server of the computer database system. *Id.* at 3:27-40, 4:14-6:13.

41. At 230, alternative reconstructed queries may also be determined, such as alternative full queries “baseball” and “basketball” when partial query “b” is submitted. *Id.* at 3:10-59. Alternative queries may be presented to the user before proceeding to the search. *Id.*

42. A search is then performed using the reconstructed full query (240) and the results of the search are presented to the user (250). *Id.*

43. Claim 1 of the ’317 Patent recites a computer database system for providing search results whose operation generally follows the sequence of Figure 2. The remaining independent claims 8 and 15 recite methods of query reconstruction that also generally follow the sequence of Figure 2.

## **B. Overview of the Prosecution History**

44. I understand that the ’317 patent is part of a family of related patents including U.S. Patent No. 7,716,201. I have reviewed the file histories of the ’317 patent and the ’201 patent as part of forming the bases of my opinions offered in this declaration. *See* EX1002 (’317 patent’s file history); EX1005 (’201 patent’s file history).

45. During prosecution, the claims of the ’317 patent were rejected for non-statutory double patenting over U.S. Patent Nos. 8,046,347 and U.S. Patent

Application Publication No. 7,716,201 to Deepa Joshi. EX1002, 56-59 (Office Action dated January 3, 2012). The Applicant overcame the rejection by issuing a terminal disclaimer for both patents. EX1002, 71 (Response to Office Action dated April 2, 2012).

46. The applications for the '317 Patent and U.S. Patent No. 8,046,347 are continuations of the application that issued as U.S. Patent No. 7,716,201. It is noteworthy that several terms used in the claims of the '317 Patent, such as “partial query” and “submitted” were discussed in the underlying prosecution of U.S. Patent No. 7,716,201.

47. US Patent No. 7,716,201 ultimately issued on May 11, 2010. During the prosecution of U.S. Patent No. 7,716,201, the claims were rejected as anticipated by U.S. Patent Application Publication No. 2006/0106769 to Gibbs (“Gibbs”). EX1005, 71-75 (Office Action dated June 17, 2008). The Applicant responded with claim amendments and with remarks arguing that the claims are patentable over Gibbs. Notably, the Applicant stated that:

“[a]s used in the present application, ‘partial search queries are shorthand ways of expressing typical search queries.’ Paragraph [0019]. . . . Gibbs, in contrast, relates to a device at a user client (202) which tries to guess what search a user is entering. The ‘search assistant’ 204 monitors user input and looks for either a ‘final input’ or a ‘partial input.’ A partial input is identified by detecting a delimiter such as a space or slash(/), number of characters or a pause in entry (para. [0033] - [0038]. Once the partial input is identified at the user client (202) it is communicated to the search engine (208). The search engine returns a set of predicted search queries. Para. [0039] In one embodiment, the search engine may return predicted search results which are

based on the predicted search queries. Para. [0044] However, Gibbs fails to disclose receiving a partial query submitted at a user client by a user seeking search results matching the submitted partial query, determining a full query based on the partial query and receiving and presenting results that match the partial query submitted by the user as in claim 1. Claims 9 and 15 have been amended to include similar features. In this regard, the processing according to claim 1 is more like Gibbs' processing of a final input, described in para. [0033] - [0034]. A final query is identified by the user entering a carriage return or selecting a search button in a user interface. In such a case, the user is submitting the final query, looking for results that match the final query.”

EX1005, 89-90 (Response to Office Action dated September 17, 2008).

48. The Examiner responded by rejecting the claims as obvious over Gibbs in view of Kamvar. EX1005, 96-101 (Office Action dated December 17, 2008). The Applicant stated, in its response, that:

“Gibbs fails to show determining a full query based on a received partial query, as the term partial query is used in this application. Since a partial query is intended to mean a shorthand way of expressing a typical search query, the partial query is not a guess at what the user is entering but it is, in fact, what the user entered and submitted by entering a carriage return or hitting "send" etc. The user's act of submitting the search query is a conscious and deliberate act and is different from the Gibbs situation of a user entering text and a software routine guessing at the search terms the user is entering.”

EX1005, 115 (Response to Office Action dated April 15, 2009).

49. The Examiner made this obviousness rejection final. EX1005, 122 (Office Action dated June 8, 2009). In its response, the Applicant amended its claims and restated the points of distinction it had previously made regarding the disclosures in Gibbs and Kamvar:

“As noted in the response filed April 15, 2009, in the presently claimed invention, a partial query is not a guess at what the user is entering, as in

Gibbs. Rather, a partial query is, in fact, what the user entered and submitted- for example by entering an abbreviated text string and then entering a carriage return or 'send' etc. The user's act of submitting the search query is a conscious and deliberate act by the user-an abbreviated query is entered, for example on a mobile device such as a cell phone.[...] In Gibbs and Kamvar, as discussed above, the disclosed system detects a query as it is being entered by the user, guesses at a set of predicted queries and interrupts the user by presenting the predicted queries to the user. The user may then select one of the predicted queries. In Gibbs and Kamvar, first, the user does not submit a partial query, as in claim 9. Rather, the user submits a query, stops upon being interrupted by the presentation of the predicted queries, then selects one of the predicted queries. No 'submission' of a partial query occurs in this case- rather, the detection of the partial query occurs when certain conditions are met, such as a detected delimiter or detecting entry of a predetermined number of characters or a timeout condition. The user does not get to 'submit' a partial query. Rather, the disclosed system decides when it is a 'partial query' and acts accordingly to identify predicted queries. Second, in the Gibbs and Kamvar system, the 'partial query' cannot be described as one for which a user is seeking search results matching the submitted partial query. [...] In effect, the user of the Gibbs and Kamvar system is typing his search query and is interrupted by the system responding with predicted queries. The partial query decided on by the system may or may not match the query for which the user seeks search results. As one example, the user may type in "ball" and the system responds with predicted queries like 'ballplayer' and 'ballgames,' when actually the user wanted search results about 'ballet.' The claim 1 act of receiving a partial query is not shown by Gibbs and Kamvar."

EX1005, 144-146 (Response to Office Action dated August 10, 2009).

50. The Applicant then issued a Request for Continued Examination wherein it restated its amendments and arguments. EX1005, 154-163 (Amendment and Request for Continued Examination dated October 8, 2009). The Examiner subsequently issued a notice of allowability for US Patent No 7,716,201 on December 24, 2009. EX1005, 174-182.

## VI. INTERPRETATION OF CLAIM TERMS

51. As discussed above, I have been informed that for the purposes of inter partes review, the standard for claim construction of terms within the claims of the patent is the same as that applied in federal district court litigation. I understand that a district court has previously construed terms of this patent, and I have been informed of those constructions. Specifically, I understand the district court in the American Airlines Litigation to have issued the following constructions:

| <b>Claim Term</b>                                                                          | <b>Claim(s)</b>      | <b>District Court's Construction</b>                                                                                                                                       |
|--------------------------------------------------------------------------------------------|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| "submitted"                                                                                | 1, 2, 4, 5, 8, 11-14 | "submitted by a conscious and deliberate act of submission (for example by entering a carriage return or by pressing 'enter' or 'send') rather than just by entering text" |
| "query data management server"                                                             | 4, 5                 | plain and ordinary meaning                                                                                                                                                 |
| "storing in memory data about previously received queries at the computer database system" | 8                    | storing, in memory, data about previously submitted queries that have been received at the computer database system"                                                       |
| "better representative of the entire search query desired by the user"                     | 1, 8                 | plain and ordinary meaning                                                                                                                                                 |
| "not fully representative of an entire search query desired by the user"                   | 1, 8                 | plain and ordinary meaning                                                                                                                                                 |

EX1011, 48-63.

52. I have applied these constructions in my analysis. For terms that the district court did not previously construe, I have generally applied the ordinary and customary meaning as understood by a POSITA in light of the specification and the prosecution history.

## **VII. OVERVIEW OF THE PRIOR ART**

### **A. Background and State of the Art for the '317 Patent**

53. At the time the '317 patent was filed, search engines and their functionalities were well known to a POSITA. In fact, all of the concepts disclosed in the '317 patent were known well before the patent's priority date. For example, several references (cited in this Petition and/or during prosecution, as discussed above in section V.B.) evidence that companies such as Microsoft and Google, had already disclosed and implemented search engines that reconstructed full queries from users' partial queries. A typical implementation would enable the user to type out a partial query then hit enter or click a "search" button on the webpage to submit the query to a server across a network, *e.g.*, the Internet. For instance, a 2003 article by Peter G. Anick (the first inventor in the Anick reference) shows that the AltaVista search engine provided users with reconstructed full queries, denoted by "feedback terms" (*e.g.*, "general election" and "election results") based on the submission of a partial query (*e.g.*, "elections").

Figure 1. Top of AltaVista search results page, showing feedback terms for the search “elections”.

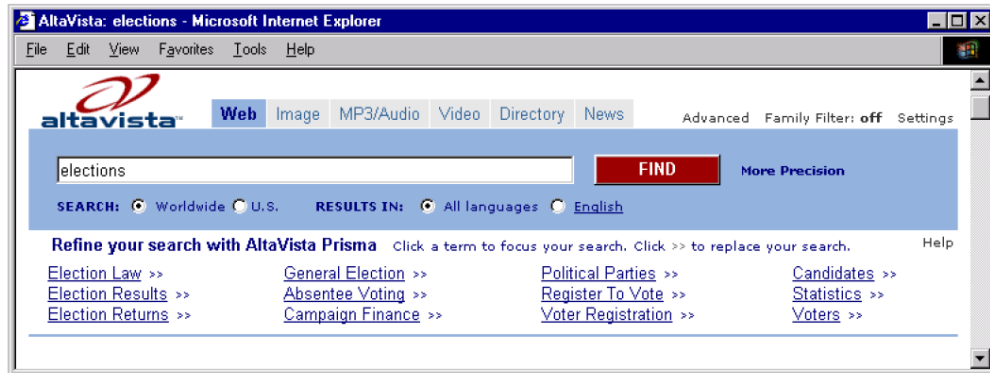
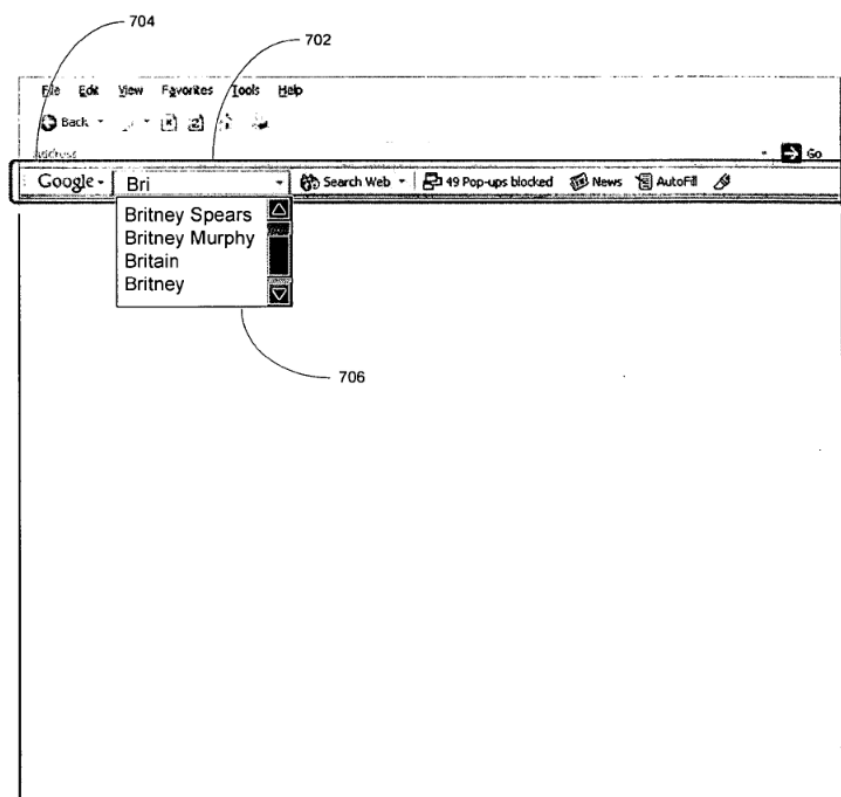


Figure 2. Top of AltaVista search results page, showing the search box and feedback terms presented after the user has clicked on the feedback term “campaign finance”.



EX1012, 89.

54. Under another well-known approach, the server would obtain partial queries as the user entered keystrokes in a search field (see Kamvar, Figure 7) and provide a dropdown list of possible reconstructed queries (e.g., reconstructing “Bri” as “Britain” or “Britney” in the example below).



Kamvar, Figure 7.

55. In either case, the search engine would match full queries to the user's partial query and perform a search using a selected reconstructed full query.

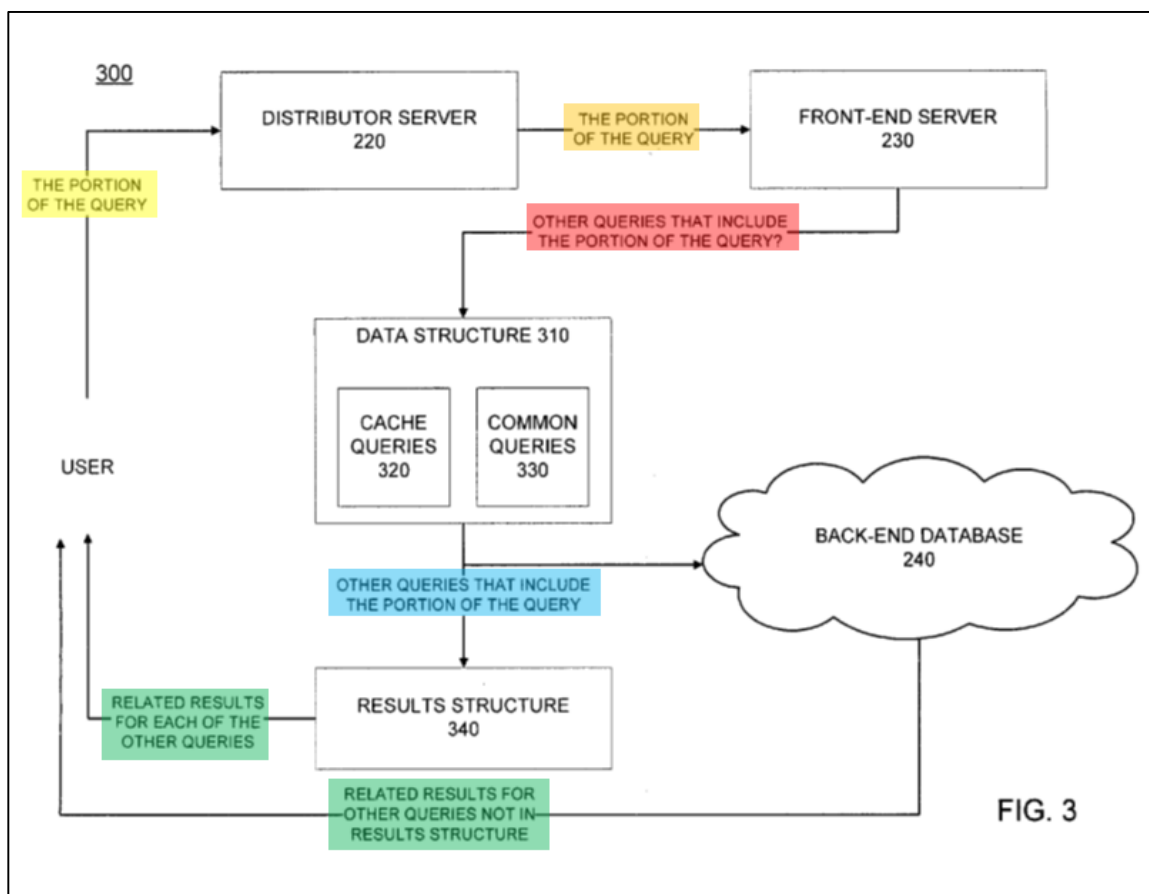
## **B. Overview of Sareen**

56. Sareen is a published patent application that was filed on February 28, 2006. It was assigned to Microsoft and lists Gaurav Sareen, Michael Isard, Yinzhe Yu and Girish Kumar as inventors.

57. Sareen discloses "a system and method for obtaining related results for a portion of a query and for generating an updated set of queries for a cache of a

server [...] Other queries beginning with the portion of the query may be identified and obtained from a data structure that includes a server cache and a set of common queries. Once the other queries are identified, results for the other queries are obtained from the server cache or from a back-end database.” Sareen, Abstract.

58. Sareen discloses various techniques to determine “updated” full queries based on the “portion” of the query entered by the user, provide the updated queries to the user and run searches using the updated queries. Figure 3 illustrates a query system implementing techniques disclosed in Sareen.



Sareen, Fig. 3 (annotated).

59. A user submits a portion of a query via the internet. A distributor server 220 receives that portion of a query. Sareen, [0029]. The distributor server distributes the query portion to one or more front-end servers 230. *Id.*, [0030].

60. Once it receives the portion of the query, the front-end server is responsible for searching for “other queries” that include the query portion. The front-end server interrogates a data structure 310 to determine whether it contains any qualifying other queries. The data structure may be part of the front-end server or separate from that server. *Id.*, [0031]. The data structure contains a cache of recent queries and search results 320, as well as a set of “common queries” 330 (“common” queries may include frequently requested queries over a predetermined time period). *Id.*, [0032]-[0036].

61. The front-end server further consults a results structure 340 that stores search results for the other queries obtained from the data structure. The results structure may be part of the front-end server or distinct from it. *Id.*, [0034]. When the front-end servers does not find those search results in the results structure, it consults a back-end database 240, which contains stored search results and an index of web pages. *Id.*, [0028], [0035].

### C. Overview of Anick

62. Anick is a U.S. patent filed on April 25, 2003 and issued on September 20, 2005 to Overture Services, Inc., a search engine company that was reportedly acquired by Yahoo! later in 2003. EX1020 (July 2003 CNET article <https://www.cnet.com/tech/tech-industry/yahoo-to-buy-overture-for-1-63-billion/>). It lists Peter G. Anick, Alastair Gourlay and John Thrall as its inventors and its title is “Systems and Methods for Interactive Search Query Refinement.” Anick, Face.

63. Anick discloses techniques for refining queries submitted by users to a search engine. Figure 2 illustrates Anick’s disclosure of a search engine server’s provision of search refinement suggestions (e.g. “Space Shuttle Challenger”) in response to an initial query (e.g. “space shuttle”).

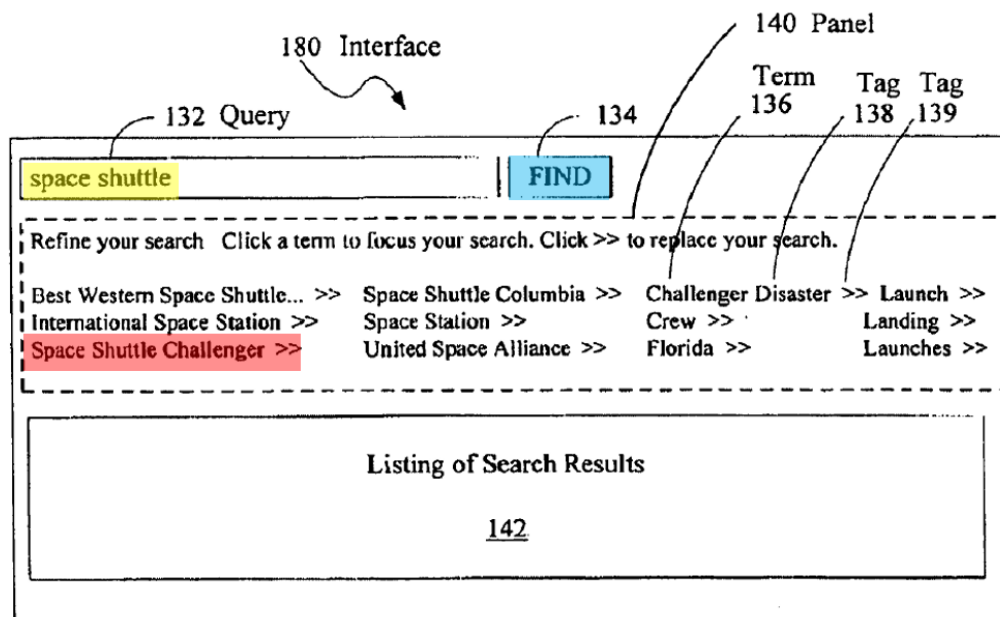


Fig. 2

Anick, Fig. 2 (annotated).

64. First, a user types an initial query 132 at a client computer and submits it to a search engine server by pressing the find button 134. Anick, 7:63-64. The server then processes the received query to provide both search results 142 and a set of candidate terms 136, each term being a search refinement suggestion that can improve the initial query. *Id.*, 7:65-8:7. The candidate terms are presented with two types of tags shown as 138 and 139. The user may click on a tag 138 to append the term to the initial query (for instance, creating the refined query “space shuttle Challenger Disaster” from the initial query “space shuttle”) and run the search on the refined query. Clicking on a tag 139 replaces the initial query with the candidate term (e.g. the search engine will then conduct a search for “Challenger Disaster”). *Id.*, 8:10-16. “Addition of the term ‘Challenger Disaster’ to the initial query or replacement of the initial query with the term ‘Challenger Disaster’ provides the user with a query that quite possibly more closely matches the interests of the user.” *Id.*, 8:40-44.

65. Anick further discloses architectures for a search engine server, as illustrated by Figure 3.

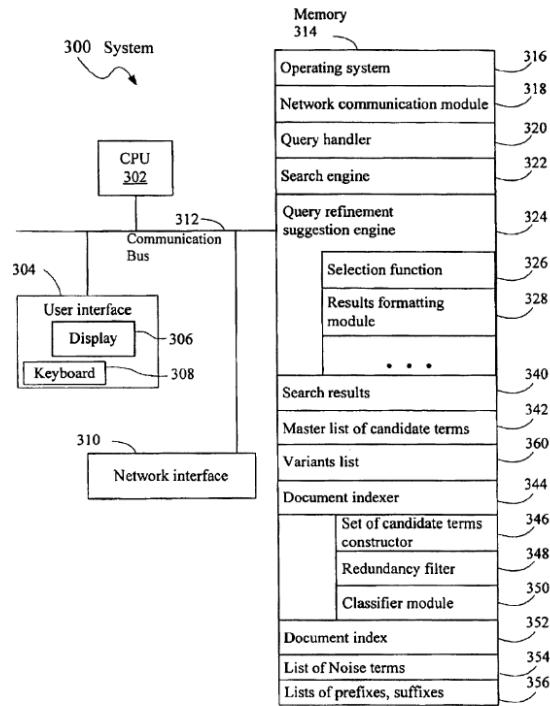


Fig. 3

66. The server includes: a query handler 320 for receiving a query from a client computer 100; a search engine 322 that searches a document index 352 for documents related to a query; and a query refinement suggestion engine 324. The query refinement suggestion engine 324 comprises a selection function 326, which identifies candidate terms for query refinement, and a results formatting module for presenting the candidate terms. *Id.*, 9:10-41.

#### D. Overview of Aravamudan

67. Aravamudan is a published patent application filed on October 7, 2005 and issued on February 21, 2012 to Veveo LLC. It lists Murali Aravamudan, Rakesh Barve, Sashikumar Venkataraman and Ajit Rajasekharan as inventors and its title is

“Method and system for incremental search with reduced text entry where the relevance of results is a dynamically computed function of user input search string character count.”

68. Aravamudan discloses methods and systems for processing web search queries that are entered as acronyms or prefixes delimited by spaces or other separators. Aravamudan, [0010]. For example, one of the disclosed search processes is illustrated in Figure 4.

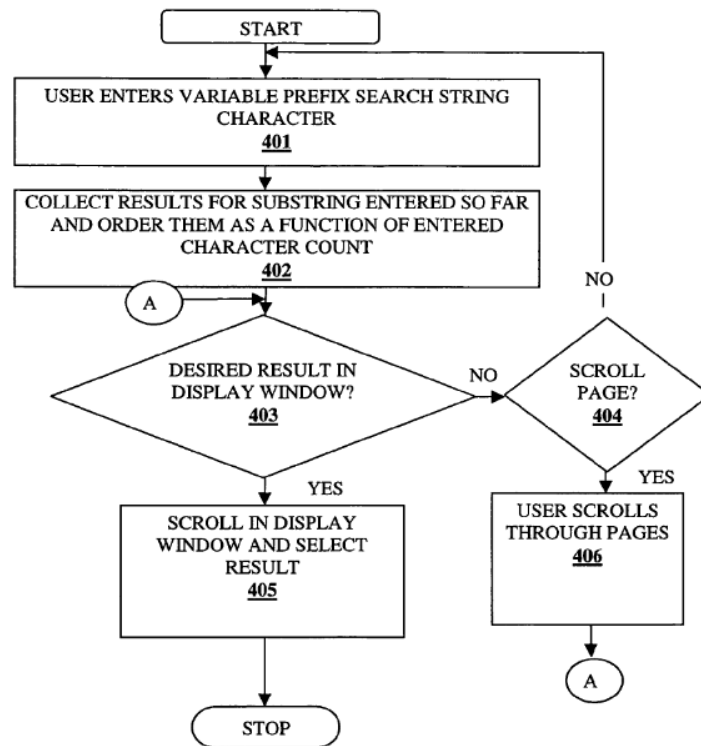


**FIG. 4**

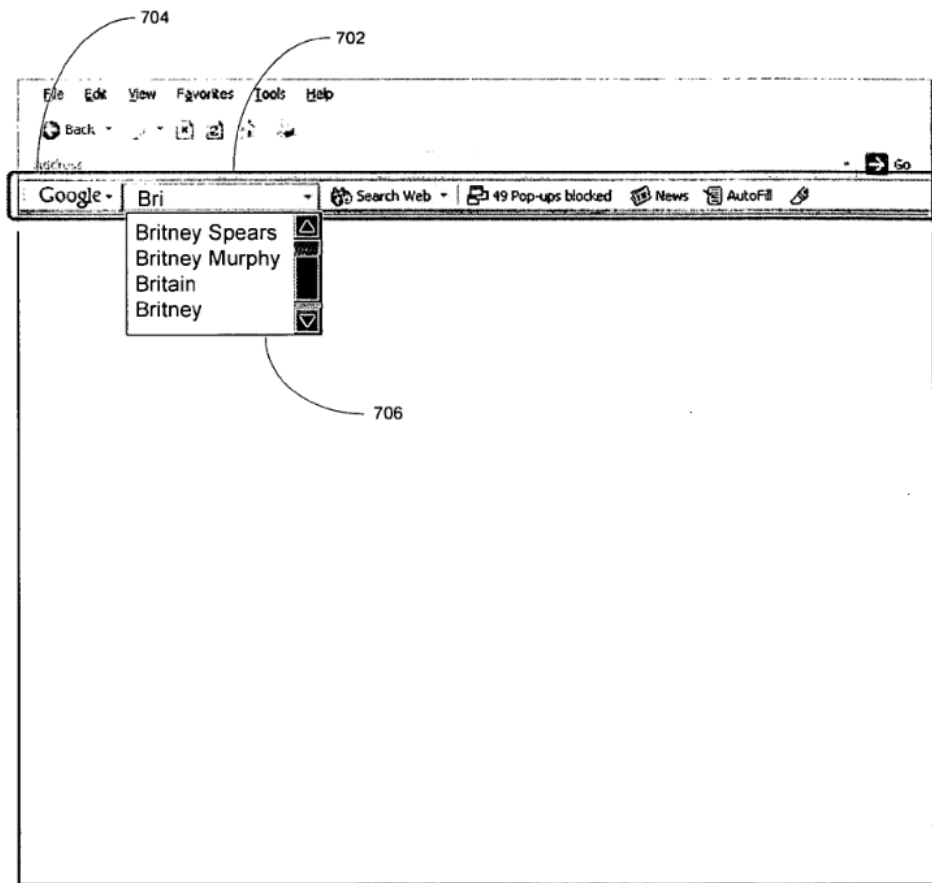
69. A user may start the process by entering a search string comprising prefixes for the “intended query at 401. For example, to represent ‘Brad Pitt,’ the

user may enter B P, BR P, B PI etc.” *Id.*, [0033]. A server system receives that search string (e.g., “B P”) and retrieves search results, at 402, by matching each delimited substring to relevant terms that may form the intended queries. The system searches for the results of the matched terms and presents the results in a display window 403 by order of relevance. The user may then select “the desired result” from the display (according to steps 403 and 405) or scroll through the pages if the desired result does not appear in the display (under steps 404 and 406).

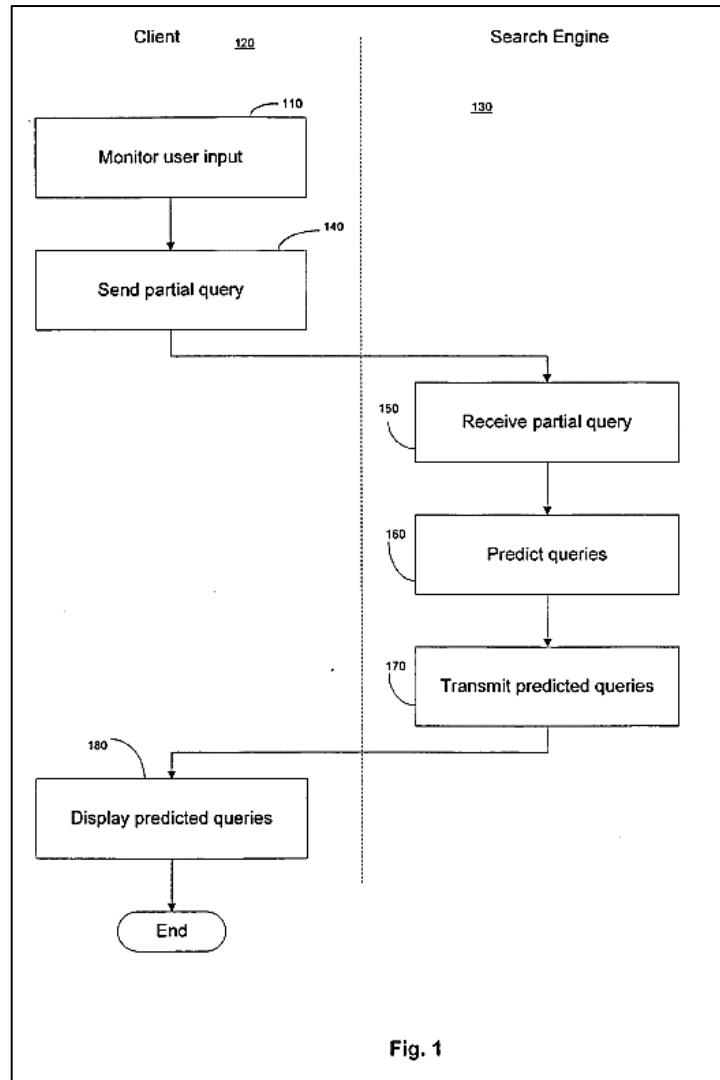
#### **E. Overview of Kamvar**

70. Kamvar is a published patent application filed on June 22, 2004 and issued on November 16, 2010, to Google LLC. It is titled “Anticipated Query Generation and Processing in a Search Engine” and lists Sepandar Kamvar, Taher Haveliwala and Glen Jeh as inventors.

71. Kamvar discloses methods and systems for monitoring a user’s input of a search query, sending the input as a partial query to a server before the user finishes entering the query and determining, at the server, a set of predicted queries based on the partial query. The user may then select a predicted query to run the search. Kamvar, [0005]. Figure 7 provides an illustration of a search engine providing predicted search queries determined from a partial query.



72. An exemplary process for the query prediction is depicted in Figure 1:



73. Kamvar discloses that partial queries may be matched to predicted queries using a dictionary listing term portions. The dictionary may be created from previously submitted search queries and may track the popularity over time of each term portion. *Id.*, [0039-42].

**VIII. GROUND 1: CLAIMS 1-11 ARE OBVIOUS IN VIEW OF SAREEN IN COMBINATION WITH ANICK.**

**A. Claim 1**

**1. [1p] Preamble**

74. The preamble of claim 1 recites “[a] computer database system for providing search results to a user in response to user submissions over a data network.” To the extent the preamble is limiting, the combination of Sareen and Anick renders this limitation obvious.

75. Sareen discloses an exemplary “computer database system” comprising “a distribution server 220 for receiving a portion of a query,” “a front-end server 230,” “a data structure 310,” a “results structure 340” and a “back-end database 240.” Sareen, [0029]-[0031], [0034]-[0035]; *see also id.*, Fig. 3 (illustrating an exemplary system).

76. A skilled artisan would understand that Sareen’s computer database system “*provid[es] search results to a user in response to a user submission.*” For example, Sareen discloses “a system and method for obtaining results for a portion of a query and for generating an updated set of queries for a cache of a server.” Sareen, Abstract. Sareen likewise discloses that its system “may be utilized to provide search results for queries, such as queries or portions of queries entered in an internet search engine through, for example, the computer 110.” Sareen, [0025].

77. A skilled artisan would understand that the user submissions in Sareen are “*over a data network.*” For example, Sareen discloses that its computer system “may operate in a network environment using logical connections to one or more remote computers,” such as a remote computer used to submit the user queries. Sareen, [0022]; *see also id.* at [0023] (discussing the operation of Sareen’s computer database system in a LAN networking environment). Sareen explains that the query may be “entered at a search engine,” which a skilled artisan would understand refers to being submitted over a data network. Sareen, [0025].

## **2. [1a] Database**

78. Claim 1 recites “*a database configured to store information about events in the computer database system.*” The combination of Sareen and Anick renders this limitation obvious.

79. Sareen discloses that “[a] cache, or local storage, may contain queries and their corresponding results.” Sareen, [0027]. More specifically, Sareen discloses that “the cache’s queries and results may be a set of recent queries and results, such as those from a predetermined time period, for example three hours.” Sareen, [0027]. Sareen further discloses storing in the local storage, “a set of common queries 330.” Sareen, [0032].

80. I understand that Patent Owner has stated in its claim construction briefing in the American Airlines Litigation that a database configured to store

previously submitted queries is sufficient to satisfy this claim limitation. EX1013, 26. A skilled artisan would understand Sareen's disclosure of storing previously submitted queries, such as common queries or recent queries, and the corresponding search results to teach the claimed "*database configured to store information about events in the computer database system.*"

81. To the extent it is found, contrary to my opinion, that Sareen does not explicitly disclose storing information about events in the computer database in a "*database*" (*i.e.*, that Sareen's local storage is not a *database*), it would have been obvious to modify Sareen to store this information in a database for at least three reasons. First, Sareen already discloses storing this information in local storage. *See* Sareen, [0027], [0034]. Second, Sareen discloses that the local storage could be located, and therefore that the information could be stored, in a number of places in the system. For example, Sareen discloses that the local storage could be a part of either the front-end server or the data structure 310. Sareen, [0027], [0031]. Third, Sareen discloses flexible persistence of its local storage, ranging anywhere from three hours to at least one week, and discloses that the storage includes "common queries" as well as "cache queries". Sareen, [0027], [0037]-[0038].

82. A skilled artisan would have been motivated to modify Sareen to store the information about previously submitted queries in a database to increase the persistence of this information, thereby extending the user's access to it. A skilled

artisan would understand that, by storing the information in the database, that information would be available to a user until it has been deleted or otherwise removed from the database. For example, storing the information in the database would ensure that the user has access to the information across multiple sessions, including after a reboot which would be useful in order to maintain the described “common queries.” Sareen discloses that storing the information locally allows for “easy and fast access to recent queries.” Sareen, [0032]. A skilled artisan would understand that benefit to exist even if the information is stored in a front-end database, because it would still be more efficient and less time-consuming to consult the front-end database “than it is to consult the back-end database, which contains a full web index and is typically larger than the front-end cache.” Sareen, [0001].

83. Saving this information in a database would have been within the routine abilities of a skilled artisan. Databases were well-known and commonly used across a very wide variety of computer systems. For example, I used the Microsoft Access database system on Windows myself since the mid 1990s. Sareen discloses storing queries and query results in multiple places, such as a front-end cache and back-end database. Sareen, [0001], [0027], [0031]-[0033]. A skilled artisan would understand that the information could also be stored in another database, such as a front-end database.

### **3. [1b] Query Reconstruction Server**

84. Claim 1 further recites “*a query reconstruction server in data communication with the database and operative to receive a partial query submitted at a remote user client system by a user seeking search results matching the submitted partial query and, in response to the received partial query, determine a full query based on: (i) the received partial query; and (ii) information stored in the database about queries previously-submitted by users.*” The combination of Sareen and Anick renders this limitation obvious.

#### **a. Data Communication with the Database**

85. Sareen discloses “*a query reconstruction server in data communication with the database*” as recited in Claim 1.

86. For example, a skilled artisan would understand Sareen’s “front-end server” to be an example of a “*query reconstruction server*” recited in Claim 1. Sareen is directed to a system for obtaining results for a portion of a query “by identifying other queries beginning with the portion of the query and obtaining the results of the other queries,” *i.e.*, by reconstructing the partial queries. Sareen, [0005]. Sareen discloses a distributor server that receives a portion of a query and transmits the partial query to the front-end server 230. Sareen, [0029]-[0030]. “Once the front-end sever 230 receives the portion of the query, the front end-server may be responsible for performing a search in order to obtain other queries that have

the same portion as the portion of the query.” Sareen [0030]. This disclosed functionality of Sareen’s front-end server mirrors the disclosed functionality of the claimed “*query reconstruction server*.” See ’317 patent at 3:64-4:1 (describing the “*query reconstruction server*” as “reconstruct[ing] a fully query from a partial query submitted via the client system 110.”).

87. A skilled artisan would understand Sareen’s front-end server to be in data communication with Sareen’s local storage (*i.e.*, “*database*”). For example, Sareen discloses that the front-end server consults the data structure 310, which in one embodiment includes the stored query data, to obtain the other queries. Sareen, [0031], [0032], [0034], Fig. 3. A skilled artisan would understand that Sareen’s disclosure of the front-end server consulting with local storage teaches “*data communication with the database*,” at least in view of Sareen’s disclosure that the data structure and the corresponding storage of query data would be part of a separate component. For example, Fig. 3 shows that the front-send server 230 performs queries to the data structure 310. Sareen, [0034].

88. Alternatively, a skilled artisan would understand that, as modified above in ¶¶81-83, Sareen’s computer database system would include a front-end server in data communication with a database. It would have been a routine modification to consult (*i.e.*, be in “*data communication with*”) a separate database

(e.g., a front-end database) storing information about previously submitted queries, as opposed to consulting the local storage (if deemed not to be a database) in Sareen.

**b. Operative to Receive a Partial Query Submitted by a User**

89. Sareen discloses that its front-end server is “*operative to receive a partial query submitted at a remote user client system by a user seeking search results matching the submitted partial query*” as recited in Claim 1. Alternatively, this limitation is rendered obvious by Sareen in view of Anick.

90. Sareen discloses a “distributor server 220 for receiving a portion of a query,” such as “the first three characters of a query” (*i.e.*, “*a partial query*”). Sareen, [0029]; *see also* Sareen, [0003]. The distributor server assigns the portion of a query to a “front-end server 230” and “may transmit the portion of the query to the assigned front-end server 230.” Sareen, [0030]. Sareen discloses that the “front-end server 230 receives the portion of the query” from the distributor server 220. Sareen, [0031]. Therefore, a skilled artisan would understand that Sareen’s front-end server (*i.e.*, the “*query reconstruction server*”) is “*operative to receive a partial query.*”

91. A skilled artisan would understand that the partial query received by the front-end server was “*submitted . . . by a user seeking search results matching the submitted partial query.*” Sareen discloses that a “user, who desires results to the query, may type or enter a portion of the query in an internet search engine.”

Sareen, [0029]. Sareen likewise discloses that a query or partial query “originates at the internet 210, where a user may insert a query for which they desire to obtain relevant results, such as web pages directed to the query.” Sareen, [0025]; *see also* Sareen, [0003].

92. A skilled artisan likewise would understand that the partial query is “*submitted at a remote user client system.*” Figure 2 of Sareen, for example, illustrates “a server organization 200 that may be utilized to provide results for queries, such as queries or portions of queries entered in an internet search engine through, for example, the computer 110.” Sareen, [0025]. A user may enter commands and information, such as queries or portions of a query, “into the computer 110 through input devices such as a keyboard 162.” Sareen, [0021], [0025].

93. I understand that the district court in the American Airlines Litigation has construed “submitted” to mean “submitted by a conscious and deliberate act of submission (for example by entering a carriage return or by pressing ‘enter’ or ‘send’) rather than just by entering text.” EX1011, 53. To the extent the Board finds that Sareen’s disclosure of a portion of a query entered by a user does not teach “*a partial query submitted*” as that phrase has been construed by the district court, it would have been obvious to a person of ordinary skill in the art to modify Sareen to incorporate that functionality based on at least Anick’s disclosures.

94. Anick discloses processing a search query that has been “submitted by a client computer 100 to a search engine server.” Anick, 7:43-44. Anick discloses an example of this submission, explaining that “when [the] find button 134 is pressed, query 132 is sent from client computer 100 to search engine server 110.” Anick, 7:63-64. “Upon receiving query 132, search engine server 110 processes received query 132 and sends search results and search refinement suggestions back to client computer 100.” Anick, 7:65-67. A skilled artisan would understand Anick’s disclosure to teach a “*partial query submitted*,” under the district court’s construction.

95. A skilled artisan would have been motivated to combine Sareen’s computer database system with Anick’s teaching of processing a partial query submitted by a user with a reasonable expectation of success to comport with user expectation in using a search engine and to achieve efficiency benefits associated with delaying processing until the user consciously submits the query.

96. To the extent Sareen is found to disclose processing user queries as the user types, but not processing those queries upon a conscious and deliberate act of submission, a skilled artisan would have been motivated to modify Sareen, at least in view of Anick, to process such queries. A user entering a search term in a search engine would expect that hitting enter, or otherwise deliberately submitting a partial query, would not reverse any processing of the query that had taken place

dynamically as the user typed the query. To the contrary, a user would expect that the act of hitting enter would not alter the functionality of the system in any way, except to make their intention clear – perhaps increasing the priority of the search request and re-processing the query.

97. Alternatively, a skilled artisan would have been motivated to modify Sareen to process queries only after they had been deliberately submitted, as opposed to processing queries dynamically as the user types. A skilled artisan would have been motivated to make this modification to more efficiently use bandwidth. Sareen’s disclosures focus on resource conservation, discussing the use of the front-end cache to preserve resources. Sareen, [0001] (“[I]t is more efficient and less time-consuming to consult the front-end cache.”). Specifically, Sareen states that its “portion-querying mechanism would be further desirable if resources were preserved and operations completed faster. . . .” Sareen, [0004]. Modifying Sareen to process queries only upon a deliberate act of submission, as opposed to dynamic processing, would have been a natural extension of Sareen’s desire to preserve resources such as reducing bandwidth by avoiding having to communicate with the back-end database. Additionally, making this modification would have reduced the computational burden on the server as it would not need to respond to as many queries. A skilled artisan would have been motivated to make this modification in order to save costs and increase server response speed by reducing this

computational burden. Given that Sareen discloses that the system load has to be considered when deciding when to perform certain kinds of queries, a skilled artisan would have been specifically motivated to take this into account. Sareen, [0049].

98. A skilled artisan would understand that by waiting until a user has made a conscious or deliberate act signifying the submission of a query, Sareen's system would be able to conserve bandwidth and other resources. This is particularly true in view of Sareen's disclosure that the user does not always want to search the first few characters of its queries (*i.e.*, "www," "http"). *See* Sareen, [0039]. A skilled artisan therefore, would be motivated to modify Sareen to wait for the user to consciously and deliberately submit the query prior to processing the query to determine a fully query, as Anick discloses.

99. Sareen mentions that it would be "desirable for the results to automatically adjust" as a user continues to type. Sareen, [0003]. A skilled artisan would have understood this disclosure to indicate a routine design choice (between processing queries in real-time and waiting until a user consciously submits the query), involving an evaluation of the trade-offs. On one hand, Sareen mentions there are benefits to processing a partial query as the user types. Sareen, [0003]. On the other hand, a skilled artisan would recognize that there would be efficiency benefits in waiting to process the query until the user commits a deliberate and conscious act of submission. *See, e.g.*, EX1014, 13:60-63 ("To conserve processing

resources or for other reasons, a limit can be set that provides a maximum number of times that a particular query can be modified or otherwise refines an update algorithm.”). Each choice has relative advantages and tradeoffs, and one skilled in the art would have recognized them and weighed them in choosing how to implement the combined system. This type of choice was common when designing a system, and one skilled in the art would have routinely made such decisions.

100. A skilled artisan would have had a reasonable expectation of success in making this combination because waiting to process a query until it has been deliberately submitted by a user was a conventional technique at the time the ’317 patent was filed, *see e.g.*, Anick at 7:43-44. A skilled artisan would have found it within their ordinary abilities to modify Sareen’s system to incorporate Anick’s disclosure of processing submitted queries since it would have been simpler. When queries are made after every keystroke, then the skilled artisan would have to take into account that the responses may come back slower than a user can type. Because of this, the system would have to be designed in order to accommodate that. I built such a system myself in the mid-2000s which is how I know that this issue would have needed to be designed for. On the other hand, if queries were only made when a user deliberately submitted them, they would be less frequent than if made after every keystroke, and thus the system would be simpler to design.

**c. Operative to Determine a Full Query**

101. Sareen discloses that “*in response to the received partial query,*” the front-end server will “*determine a full query based on: (i) the received partial query; and (ii) information stored in the database about queries previously-submitted by users*” as recited in Claim 1.

102. Sareen’s system is directed to returning relevant results in response to a partial query, as opposed to a “full search term or query.” Sareen, [0003]. Sareen discloses that a portion of a query is received and compared against previously submitted queries to determine full queries that begin with that portion of a query. Specifically, Sareen explains that “[o]nce the front-end server 230 receives the portion of the query, the front-end server 230 may be responsible for performing a search in order to obtain other queries that have the same portion as the portion of the query.” Sareen, [0031]. A skilled artisan would understand Sareen’s disclosure of “obtain[ing] other queries that have the same portion as the portion of the query” to mean that Sareen’s front-end server “*determine[s] a full query.*”

103. A skilled artisan would understand that Sareen’s front-end server determines the full query based on “*(i) the received partial query; and (ii) information stored in the database about queries previously submitted by users.*” For example, Sareen discloses that in determining full queries, the front-end server “may consult a data structure 310,” which may include the local storage (*i.e.*, the

“database”). Sareen, [0031]. For example, the data structure in Sareen may store both “the most recent queries” submitted by a user and a set of “frequently requested queries.” Sareen, [0031]-[0032], [0038]. In consulting the information stored in the data structure, the front-end server compares the portion of the query (*i.e.*, “*the received partial query*”) to recent queries and frequently requested queries (*i.e.*, “*queries previously submitted by users*”), thereby “*determin[ing] a full query based on: (i) the received partial query; and (ii) information stored in the database about queries previously submitted by users.*”

104. A skilled artisan would understand that the information in the data structure that is consulted by the front-end server is stored in local storage (*i.e.*, “*stored in the database*”). As discussed above, *see* Section VIII.A.2,<sup>5</sup> to the extent Sareen does not disclose that this information is stored “*in a database*,” a skilled artisan would have been motivated to modify Sareen to include a database containing this information. A skilled artisan would understand that Sareen, as modified to store the information about previously submitted queries in a database, would determine the full query in the same way as it does without modification. A skilled artisan would understand that the front-end server, upon receiving the partial query, would consult the previously submitted query data to “obtain other queries that have the

---

<sup>5</sup> As used herein, references to previous discussions should be understood to incorporate those discussions in full.

same portion as the portion of the query.” *See* Sareen, [0031]. It would have been a routine modification to consult the database as opposed to the local storage (if deemed not to be a database) in Sareen. *See* ¶88.

#### **4. [1c] Full Query Better Representative of Entire Query**

105. Claim 1 further recites “*wherein the submitted partial query comprises an abbreviated or incomplete search query which is not fully representative of an entire query desired by the user and the full query is better representative of the entire search query desired by the user.*” The combination of Sareen and Anick renders this limitation obvious.

106. Sareen discloses a “*submitted partial query*” that is a “portion of a query, such as the first few characters” (*i.e.*, “*an abbreviated or incomplete search*”). Sareen, [0003]. Sareen distinguishes the submitted “portion of a query” from a “full search term or query.” Sareen, [0003]. For example, Sareen discloses a “portion of a query” submitted by “a user, who, desiring results related, to the query, enters the portion of the query in an internet search engine for example.” Sareen, [0039]. A skilled artisan therefore, would understand that Sareen’s portion of a query (e.g., the first few characters of a query) is “*not fully representative of an entire query desired by the user.*”

107. Sareen contrasts its systems with previous systems where a user’s submission of a partial query often returned an error message or incorrect results.

Sareen, [0003]. Sareen's system instead returns relevant results for a portion of a query "by identifying other queries beginning with the portion of the query and obtaining the results of the other queries." Sareen, [0005]. A skilled artisan therefore, would understand that Sareen's "other queries beginning with the portion of the query" (*i.e.*, "*full quer[ies]*") are "*better representative of the full entire search query desired by the user.*"

**B. Claim 2**

108. Claim 2 recites "[t]he computer database system of claim 1 wherein the query reconstruction server is further operative to determine as the full query the most frequently submitted, previously-submitted query that matches partial words of the partial query." The combination of Sareen and Anick renders this limitation obvious.

109. As discussed above, Sareen discloses that in determining full queries, the front-end server "may consult a data structure 310," which may include local storage. Sareen, [0031]. The data structure in Sareen may store both "the most recent queries" submitted by a user and a set of "frequently requested queries." Sareen, [0031]-[0032], [0038]. Sareen further discloses that the information stored in the data structure may "for example be a list of a predetermined number of most frequently requested queries beginning with the portion of the query. The list may be sorted in a descending order of frequency, so that the most common requested

query beginning with the portion of the query is first listed.” Sareen, [0033]. A skilled artisan therefore, would understand that one of the full queries determined by Sareen is the “most common requested query beginning with the portion of the query” (*i.e.*, “*most frequently-submitted, previously-submitted query that matches the partial query*”).

### C. Claim 3

110. Claim 3 recites “[t]he computer database system of claim 2 further comprising: a web server operative to receive search results produced by a search engine and deliver web pages that include the received results to a browser of the user.” The combination of Sareen and Anick renders this limitation obvious.

111. Sareen discloses that its back-end database 240 may contain search results, including web pages. Sareen, [0028]. Sareen also discloses a results structure 340, which “contains results for queries” that have been generated by a search engine. Sareen, [0034]. “Upon consultation with the results structure 340, results for queries that begin with the portion of the query are obtained and provided to the user.” Sareen, [0034]. The presented results “may be, for example, a multiple selection of web pages.” Sareen, [0034].

112. A skilled artisan would understand that the results are “obtained and provided to the user” by a “web server.” For example, Sareen discloses a “distributor server” and “front-end servers” (each a “web server”) which receive a user’s search

from “a web search engine.” Sareen, [0001], [0025] (“The internet 210 may be connected to a distributor server 220, which receives the query from the internet.”). The distributor server transmits the query to a front-end server, which then obtains full queries and the corresponding search results, including web pages. Sareen, [0031]-[0034]. As illustrated in Figure 2, a skilled artisan would understand that the distributor server 220 serves as the gateway between the back-end database, front-end server, and the internet. A skilled artisan therefore, would understand that the results “which are obtained and provided to the user” are “*receive[d]*” by a front-end server and “*deliver[ed]*” to the user by way of distributor server 220. As such, both the front-end server and distributor servers are web servers because they each provide the functions that a skilled artisan would expect of a web server – namely receiving requests for web content and delivering that web content.

113. A skilled artisan likewise would understand that the results that are provided to the user are “*delivered to a browser of the user.*” Sareen discloses that the search “may be entered at a search engine, which may allow the user to insert the query of interest and which may display the relevant results for the user to access.” Sareen, [0025]. A skilled artisan would understand that the user would access the search engine through a browser, such as on a personal computer as disclosed by Sareen. “The computer 110 may operate in a networked environment using logical connections to one or more remote computers, such as a remote

computer 180. The remote computer 180 may be a personal computer ....” Sareen, [0022] (“The computer 110 may operate in a networked environment using logical connections to one or more remote computers, such as a remote computer 180. The remote computer 180 may be a personal computer. . . .”).

**D. Claim 4**

114. Claim 4 recites “[t]he computer database system of claim 1 further comprising: a query data management server in data communication with the query reconstruction server and the database, the query data management server being operative to provide to the query reconstruction server the information stored in the database about queries previously-submitted by users.” The combination of Sareen and Anick renders this limitation obvious.

115. Sareen discloses a “query data management server in data communication with the query reconstruction server and the database.” For example, Sareen illustrates a data structure 310 in data communication with the front-end server (i.e., “the query reconstruction server”) and the local storage containing information about previously submitted queries (i.e., “the database”). Alternatively, it would have been a routine modification to consult (i.e., be in “data communication with”) the database as opposed to the local storage (if deemed not to be a database) in Sareen. See ¶88. A skilled artisan would understand Sareen’s “data structure 310” to disclose the claimed “query data management server.”

116. As discussed above, *see* Section VIII.A.3.c, Sareen discloses that the front-end server “may consult a data structure 310” in determining the full queries. Sareen, [0031]. The data structure in Sareen may store both “the most recent queries” submitted by a user and a set of “frequently requested queries.” Sareen, [0031]-[0032], [0038]. When the front-end server consults the data structure, the data structure provides the front-end server (*i.e.*, “*the query reconstruction server*”) with recent queries and frequently requested queries (*i.e.*, “*information . . . about queries previously submitted by users*”). As discussed above, *see* Section VIII.A.3.c, a skilled artisan would understand that this information is “*stored in the database.*”

117. A skilled artisan would understand that Sareen’s system is flexible as to the location of the data structure. For example, Sareen discloses that the data structure 310 could be either a part of the front-end server 230 or a separate component in the system. Sareen, [0031]; *see also id.* Fig. 3. If the data structure is a separate component, as illustrated in Figure 3 of Sareen, a skilled artisan would understand that it would be included in a separate server. A skilled artisan therefore would understand that Sareen’s data structure 310 could be on its own server (*i.e.*, “*query data management server*”).

118. To the extent it is found that Sareen’s data structure is not a “server,” it would have been obvious to modify Sareen’s system to include a server to perform the functionality described with respect to Sareen’s data structure. A skilled artisan

would have been motivated to make this modification with a reasonable expectation of success at least because of the disclosed flexibility in Sareen's system. A skilled artisan would have understood that Sareen's data structure could be implemented as part of a server and would have been motivated to do so to provide better performance. *See* Sareen, [0031] (The data structure "may be part of the front-end server 230."). Separating services on different computers is one well-known way to improve performance. If the data structure is not part of the front-end server, there are only a limited number of places it could be housed, including a client-side application and a separate server.

119. Skilled artisans were familiar with the standard software engineering practice of "separation of concerns" which refers to the idea that by separating the components of a technical system and minimizing the coordination between them, each can be developed, tested, and updated separately from each other. "Separation of concerns" is at the core of software engineering and has been for decades. In its most general form, it refers to the ability to identify, encapsulate, and manipulate only those parts of software that are relevant to a particular concept, goal, or purpose. EX1015, 809-810. This decreases cost and complexity while increasing reliability. In computer systems, this principle lets you upgrade your operating system without having to change your personal documents. In this context, separating one server

from another would mean that one could be updated or maintained without impacting the other.

120. Placing the data structure on a separate server would have been within the routine abilities of a skilled artisan. As just described, separating these functions would make the software simpler and more easily maintained.

#### **E. Claim 5**

121. Claim 5 recites “[t]he computer database system of claim 4 wherein the query data management server is further configured to: retrieve from the database historical query data relating to a plurality of previously submitted search queries; build a query index based on the historical query data; store the query index in the database; and provide the query index to the query reconstruction server.” The combination of Sareen and Anick renders these limitations obvious.

122. Sareen discloses that its data structure (“query data management server”) may store both “the most recent queries” submitted by a user and a set of “frequently requested queries” (i.e., “historical query data relating to a plurality of previously submitted queries”). Sareen, [0031]-[0032], [0038]. As discussed above, see Section VIII.A.3.c, a skilled artisan would understand that this information is “retrieve[d] from the database.” The data structure is thus “configured to . . . retrieve from the database historical query data relating to a plurality of previously submitted search queries.”

123. Sareen discloses that the data structure may use the historical query data to build a query index. For example, Sareen discloses that the “data structure may be an augmented trie structure, which is a structure that facilitates fast retrieval of information related to a query or a portion of query.” Sareen, [0033]. The trie structure (*i.e.*, “*query index*”) may include “possible portions of a query along with additional information.” Sareen, [0033]. “The information related to each portion of the query may for example be a list of a predetermined number of most frequently requested queries beginning with the portion of the query.” Sareen, [0033]. A skilled artisan therefore, would understand that the data structure is “*configured to . . . build a query index based on the historical query data.*”

124. Sareen discloses that the front-end server (*i.e.*, “*the query reconstruction server*”) consults, thereby retrieving information from the data structure. A skilled artisan would understand that the information that the front-end server retrieves may include the indexed information, such as a list “of a predetermined number of most frequently requested queries beginning with the portion of the query.” Sareen, [0033]-[0034]. A skilled artisan therefore, would understanding that the data structure is “*configured to . . . provide the query index to the query reconstruction server.*”

**F. Claim 6**

125. Claim 6 recites “[t]he computer database system of claim 4 wherein the query data management server is further configured to: retrieve from the database information about queries previously submitted to the computer database; filter the retrieved information about the queries; and construct one or more indexes from the filtered query information, the query data management server providing the constructed indexes to the query reconstruction server for use as the information stored in the database about queries previously-submitted by users when the query reconstruction server determines the full query.” The combination of Sareen and Anick renders these limitations obvious.

126. As discussed above with respect to Claim 5, a skilled artisan would understand that Sareen discloses a “query data management server [] configured to: retrieve from the database information about queries previously submitted to the computer database; . . . construct one or more indexes . . . [and] provid[e] the constructed indexes to the query reconstruction server for use as the information stored in the database about queries previously-submitted by users when the query reconstruction server determines the full query.” See Section VIII.E.

127. A skilled artisan would further understand that Sareen discloses “filter[ing] the retrieved information about the queries.” As discussed above, Sareen discloses that its data structure (i.e., “query data management server”) may

be an augmented trie structure, which a skilled artisan would understand to be an existing data structure that has been augmented (*e.g.*, “*filter[ed]*”) to fit an application’s needs. In Sareen, this “*filter[ing]*” may be in the form of modifying the historical data to form a list “of a predetermined number of most frequently requested queries beginning with the portion of the query.” Sareen, [0033]. The list may be sorted in a descending order of frequency, so that the most common requested query beginning with the portion of the query is first listed. Sareen, [0033]. The data may also be filtered to include only a set of queries and results, such as those from a predetermined time period. Sareen, [0027].

128. A skilled artisan likewise would understand that the data structure’s “trie structure” (*i.e.*, “*query index*”) is constructed using the filtered list of most frequently requested queries beginning with the portion of the query (*i.e.*, “*the query data management server is configured to . . . construct one or more indexes from the filtered query information*”). See Sareen, [0033].

#### **G. Claim 7**

129. Claim 7 recites “[*t*]he computer database system of claim 6 wherein the query data management server is configured to select, as the filtered query information, most popular queries from the retrieved information about queries previously submitted.” The combination of Sareen and Anick renders this limitation obvious.

130. As discussed above with respect to Claim 6, Sareen discloses filtering the historical query data to form a list “of a predetermined number of most frequently requested queries beginning with the portion of the query.” Sareen, [0033]; *see* Section VIII.F. A skilled artisan would understand the “most frequently requested queries” to be the “*most popular queries from the retrieved information about queries previously submitted*” because popular things are performed frequently.

## **H. Claim 8**

### **1. [8p] Preamble**

131. The preamble of claim 8 recites “[a] computer implemented method for a computer database system.” To the extent the preamble is limiting, the combination of Sareen and Anick renders this limitation obvious.

132. Sareen discloses an exemplary “computer database system” comprising “a distribution server 220 for receiving a portion of a query,” “a front-end server 230,” “a data structure 310,” including local storage, a “results structure 340” and a “back-end database 240.” Sareen, [0029]-[0031], [0034]-[0035]; *see also id.*, Fig. 3 (illustrating an exemplary system).

133. A skilled artisan would understand that Sareen discloses a “computer-implemented method” for its database system. For example, Sareen discloses “a system and method for obtaining results for a portion of a query and for generating an updated set of queries for a cache of a server.” Sareen, Abstract. Sareen likewise

discloses that its system “may be utilized to provide search results for queries, such as queries or portions of queries entered in an internet search engine through, for example, the computer 110.” Sareen, [0025].

## **2. [8a] Storing in Memory**

134. Claim 8 recites “*storing in memory data about previously received queries at the computer database system.*” The combination of Sareen and Anick renders this limitation obvious.

135. Sareen discloses that “[a] cache, or local storage, may contain queries and their corresponding results.” Sareen, [0027]. More specifically, Sareen discloses that “the cache’s queries and results may be a set of recent queries and results, such as those from a predetermined time period, for example three hours.” Sareen, [0027]. Sareen further discloses storing a set of common, or frequently requested, queries. Sareen, [0032]-[0034]. Sareen discloses that recent queries and common queries “may be maintained in the data structure 310 for easy and fast access to recent queries.” Sareen, [0032]. A skilled artisan would understand that Sareen’s data structure discloses “*storing in memory data about previously received queries at the computer database system.*”

## **3. [8b] Retrieving Data from the Memory**

136. Claim 8 recites “*at a server, retrieving selected data from the memory including query data representative of previously submitted queries and information*

*based on frequency counts for the respective previously submitted queries.”* The combination of Sareen and Anick renders this limitation obvious.

137. Sareen discloses local storage that may store both “the most recent queries” submitted by a user and a set of “frequently requested queries” (*i.e.*, “*query data representative of previously submitted queries and information based on frequency counts for the respective previously submitted queries.*” Sareen, [0031]-[0032], [0038]. Sareen further discloses that the data structure 310 may comprise an augmented trie structure, including “possible portions of a query along with additional information.” Sareen, [0033]. Sareen discloses that the additional information “may for example be a list of a predetermined number of most frequently requested queries beginning with the portion of the query.” Sareen, [0033]. A skilled artisan would understand the list of frequently requested queries to be filtered based on the frequency counts of the previously submitted queries (*i.e.*, to be “*information based on frequency counts for the respective previously submitted queries*”).

138. A skilled artisan would understand that the information stored in the data structure is “*retrieved . . . from the database.*” As discussed above, *see* Section VIII.A.2 *supra*, Sareen discloses local storage configured to store information about previously submitted queries.

#### **4. [8c] Constructing Indexes**

139. Claim 8 recites “*at the server, constructing indexes using the retrieved selected data.*” The combination of Sareen and Anick renders this limitation obvious.

140. As discussed above with respect to Claim 5, Sareen discloses that the data structure may use the historical query data retrieved from the database to build query indexes (*i.e.*, “*constructing indexes using the retrieved selected data*”). See Section VIII.E. For example, Sareen discloses that the “data structure may be an augmented trie structure, which is a structure that facilitates fast retrieval of information related to a query or a portion of query.” Sareen, [0033]. The trie structure may include “possible portions of a query along with additional information.” Sareen, [0033]. “The information related to each portion of the query may for example be a list of a predetermined number of most frequently requested queries beginning with the portion of the query.” Sareen, [0033]. Sareen also discloses storing in the data structure a set of “recent queries and results, such as those from a predetermined time period, for example three hours.” Sareen, [0027], [0033].

#### **5. 8[d] Receiving a Partial Query**

141. Claim 8 recites “*receiving at the server a partial query submitted by a user at a remote user client system, the submitted partial query including an*

*abbreviated or incomplete search query which is not fully representative of an entire search query desired by the user.”* The combination of Sareen and Anick renders this limitation obvious.

142. Sareen discloses “*receiving a partial query submitted by a user at a remote user client system*” for the reasons discussed above with respect to Claim 1. See Section VIII.A.3.

143. Sareen also discloses “*the submitted partial query including an abbreviated or incomplete search query which is not fully representative of an entire search query desired by the user*” for the reasons discussed above with respect to Claim 1. See Section VIII.A.4.

144. To the extent the Board finds that Sareen’s disclosure of a portion of a query entered by a user does not teach “*a partial query submitted by a user,*” this limitation would have been obvious in view of Sareen and Anick for the reasons discussed above with respect to Claim 1. See Section VIII.A.3.

## **6. [8e] Determining a Full Query**

145. Claim 8 recites “*at the server, using the indexes to determine a full query corresponding to the submitted partial query, the full query being better representative of the entire search query desired by the user.*” The combination of Sareen and Anick renders this limitation obvious.

146. Sareen discloses “*determin[ing] a full query corresponding to the submitted partial query, the full query being better representative of the entire search query desired by the user*” for the reasons discussed above with respect to Claim 1. See Sections VIII.A.3-VIII.A.4.

147. A skilled artisan would understand that the “*determin[ation] [of] a full query*” occurred “*at the server, using the indexes to determine a full query*” for the reasons discussed above with respect to Claims 1 and 5. For example, a skilled artisan would understand that the full query in Sareen is determined by the front-end server (*i.e.*, “*at the server*”) using recent queries and frequently requested queries stored in the data structure (*i.e.*, “*using the indexes*”). See Sections VIII.A.3.c, VIII.E.

#### **I. Claim 9**

148. Claim 9 recites “*[t]he method of claim 8 further comprising communicating the full query from the server to a search engine for subsequent processing of the full query.*” The combination of Sareen and Anick renders this limitation obvious.

149. Sareen discloses that “[w]hen a list of other queries, which may include the portion of the query, is obtained [by the front-end server] from the data structure, a results structure 340 . . . is consulted.” Sareen, [0034]. Sareen discloses that the front-end server may also consult the back-end database 240 “if results for one or

more of the queries that include the portion of the query are not located in the results structure.” Sareen, [0035]. A skilled artisan would understand that the “consultation” disclosed in Sareen involves “*communicating the full quer[ies]*” obtained by the front-end server to the database or results structure.

150. A skilled artisan would understand that the data structure and back-end database are part of the search engine. A skilled artisan therefore, would understand that the full queries obtained by the front-end server in Sareen are communicated from the server to the search engine (“*communicating the full query from the server to a search engine for further processing*”).

**J. Claim 10**

151. Claim 10 recites “[t]he method of claim 8 further comprising: *submitting the full query from the server to a search engine; receiving search results at the server from the search engine; and communicating a web page including at least some of the search results from the server to the user.*” The combination of Sareen and Anick renders this limitation obvious.

152. Sareen discloses that “[w]hen a list of other queries, which may include the portion of the query, is obtained [by the front-end server] from the data structure, a results structure 340 . . . is consulted.” Sareen, [0034]. Sareen discloses that the front-end server may also consult the back-end database 240 “if results for one or more of the queries that include the portion of the query are not located in the results

structure.” Sareen, [0035]. A skilled artisan would understand that the “consultation” disclosed in Sareen involves “*submitting the full quer[ies]*” obtained by the front-end server to the database or results structure. A skilled artisan would understand that the results structure and back-end database are part of the search engine. A skilled artisan therefore, would understand that the full queries obtained by the front-end server in Sareen are communicated from the server to the search engine (“*submitting the full query from the server to a search engine*”).

153. Sareen discloses that its front-end server (*i.e.*, “*server*”) consults the results structure and back-end database (*i.e.*, part of the “*search engine*”) to obtain search results (*i.e.*, “*receiv[es] search results*”). Specifically, Sareen discloses that “[u]pon consultation with the results structure 340, results for queries that begin with the portion of the query are obtained and provided to the user.” Sareen, [0034]. Sareen also discloses that the front-end server obtains search results from the back-end database. Sareen, [0035]. A skilled artisan would understand that these results are obtained by Sareen’s front-end server (*i.e.*, “*receiving search results at the server from the search engine*”).

154. A skilled artisan would understand that Sareen discloses “*communicating a web page including at least some of the search results from the server to the user*” for at least the reasons discussed above with respect to Claim 3. See Section VIII.C.

## K. Claim 11

155. Claim 11 recites “[t]he method of claim 10 further comprising: at the server, using the indexes to determine one or more alternative queries that match the submitted partial query, and wherein communicating the web page comprises communicating a web page including at least some of the one or more alternative queries.” The combination of Sareen and Anick renders this limitation obvious.

156. A skilled artisan would understand that Sareen discloses “using the indexes to determine one or more alternative queries that match the submitted partial query” for at least the reasons discussed above with respect to Claim 8. See Section VIII.H.6. For example, Sareen discloses that a portion of a query is received and compared against indexes of previously submitted queries to determine full queries that begin with that portion of a query. Specifically, Sareen explains that “[o]nce the front-end server 230 receives the portion of the query, the front-end server 230 may be responsible for performing a search in order to obtain other queries that have the same portion as the portion of the query.” Sareen, [0031]. A skilled artisan therefore, would understand Sareen to disclose determining multiple full queries that match the portion of the query submitted by the user, one of the queries being the “full query” and the remaining queries being “one or more alternative queries that match the submitted partial query.”

157. Although Sareen discloses presenting search results for at least some of these alternative queries, it does not expressly disclose communicating a web page listing the alternative queries themselves. It would have been obvious, however, to include that functionality within Sareen's system based on the teachings of Anick.

158. Anick discloses returning to the user "a subset of candidate terms (search refinement suggestions)" (*i.e.*, "*one or more alternative queries*") along with a "listing of search results (top-ranked documents in an initial list of ranked documents)." Anick, 7:53-56, 8:2-7. For example, for the partial query "space shuttle," Anick discloses returning several alternative queries, including "Space Shuttle Challenger" and "Space Shuttle Columbia." Anick, Fig. 2. These alternative queries are displayed on the user's interface, such as the user's search engine browser (*i.e.*, are "*a web page including at least some of the one or more alternative queries*"). See Anick, Fig. 2, 8:2-3.

159. A skilled artisan would have been motivated to combine this functionality disclosed in Anick with Sareen's system to aid the user in efficiently refining his or her query to obtain more useful results. For example, Anick discloses that the "list of suggested terms [] can be used to improve an initial query." Anick, 8:34-36. Anick likewise suggests that the suggested queries may "more closely match[] the interests of the user. By using the novel subset of candidate terms, a user can build an improved query without analyzing documents (or indicia thereof)

within the initial group of ranked documents.” Anick, 8:43-47. In other words, incorporating this functionality disclosed in Anick into Sareen’s system would limit the “need to determine why an initial query produced too many (or too few) results or results that are not directly related to the informational needs of the user.” Anick, 8:47-51.

160. A skilled artisan would have had a reasonable expectation of success in making this combination. As discussed above, ¶156, Sareen’s system already determines alternative queries. Sareen’s system likewise identifies and presents results corresponding to these alternative queries. *See* Sareen, [0034]. Extending this functionality to present to the user the alternative queries themselves, as disclosed by Anick, would have been well within the ordinary abilities of a skilled artisan. This is merely a specific example of the well-known practice of making suggestions to users based on their initial input. For example, there is a whole field of having computer systems make personalized recommendations based on the similarity to the past behavior of other users, or similarity of items that a user has expressed interest in. Often called “collaborative filtering,” this was something I was aware of at least since the 1990s.

**IX. GROUND 2: CLAIMS 12-14 ARE OBVIOUS IN VIEW OF SAREEN IN COMBINATION WITH ANICK AND ARAVAMUDAN.**

**A. Claim 12**

161. Claim 12 recites “[t]he method of claim 8 wherein receiving the partial query comprises interpreting the submitted partial query as a series of words separated by delimiters.” The combination of Sareen, Anick and Aravamudan renders this limitation obvious.

162. The combination of Sareen and Anick renders obvious the limitations of claim 8 for at least the reasons discuss above. *See* Section VIII.H. To the extent the Board finds that Sareen does not “interpret[] the submitted partial query as a series of words separated by delimiters,” it would have been obvious to implement that functionality into Sareen’s system in view of at least the teachings of Aravamudan.

163. Aravamudan discloses a “user starting a new search, entering characters and arriving at the desired result.” Aravamudan, [0033]. A user “enters a search string character at 401, which could be a variable size prefix of the intended query” (i.e, “a submitted partial query”). Aravamudan, [0033]. As an example, Aravamudan discloses entering “B P”, “BR P,” “B PI,” etc. to represent “Brad Pitt.” Aravamudan, [0033]. Aravamudan discloses that the system then “dynamically retrieves results at 402 for the cumulative substring of characters the user enters and orders the collected results based on (i) the relevance of the matched terms [], and

(ii) the number of characters the user entered.” Aravamudan, [0033]. A skilled artisan therefore would understand Aravamudan to disclose interpreting the submitted query of “B [space] P” as separate words matching the full query “Brad Pitt” (*i.e.*, “*interpreting the submitted partial query as a series of words separated by delimiters*”).

164. A skilled artisan would have been motivated to combine this functionality with Sareen’s system with a reasonable expectation of success to more efficiently reconstruct a user’s query and to obtain more relevant results. Sareen already discloses processing a user’s partial query to identify “the proper portion of the query” to be searched. Sareen, [0039] (“For example, if a user enters such queries beginning with “http” and “www,” the user typically does not want to search these queries but may instead be interested in the characters following these frequently-occurring entries.”). Recognizing a user’s submission of a delimiter, such as a space, as an indication that the partial query is a series of words, as disclosed in Aravamudan, would be a natural extension of Sareen’s focus on identifying the proper partial query to be searched. Using delimiters to “tokenize” the input in a query has been a standard and well-known way to process queries for decades and is the standard way to process input in the broad field of “information retrieval.” This is, in part, because databases are indexed by the words in the data

set, and being able to access the database by the words in the query is a normal way to match up the query with the way the database is indexed.

**B. Claim 13**

165. Claim 13 recites “[t]he method of claim 12 further comprising: at the server, interpreting each word of the received partial query as a partial word of the entire search query desired by the user; and using the indexes, determining a full query that matches the partial words specified in the partial query.” The combination of Sareen, Aravamudan and Anick renders this limitation obvious.

166. Aravamudan discloses “*interpreting each word of the received partial query as a partial word of the entire search query desired by the user*” and “*determining a full query that matches the partial words specified in the partial query*” for at least the reasons discussed above with respect to Claim 12. See Section IX.A. For example, Aravamudan discloses interpreting the submitted query of “B [space] P” as separate partial words matching the full query “Brad Pitt.” Aravamudan, [0033].

167. A skilled artisan would understand that Aravamudan’s disclosed functionality of “*determining a full query that matches the partial words specified in the partial query*” would be implemented in Sareen using Sareen’s indexes (*i.e.*, “*using the indexes*”). As discussed above with respect to Claim 8, Sareen discloses constructing indexes and using the indexes to determine a full query corresponding

to the submitted partial query. *See* Section VIII.H. A skilled artisan would understand that full queries matching multi-word partial queries, such as those disclosed in Aravamudan, would also be determined using the constructed indexes disclosed in Sareen because that is a known function of an index in computer database systems such as those disclosed in Sareen or Aravamudan.

168. A skilled artisan would have been motivated to combine Sareen and Aravamudan for the reasons discussed above with respect to Claim 12. *See* Section IX.A.

**C. Claim 14**

169. Claim 14 recites “[t]he method of claim 13 wherein determining a full query comprises determining the full query that is the most frequently submitted known query, as defined by the indexes, that matches the partial words of the received partial query in the same order.” The combination of Sareen, Aravamudan, and Anick renders this limitation obvious.

170. As discussed above with respect to Claim 2, Sareen discloses “*determin[ing] as the full query the most frequently submitted, previously-submitted query . . . that matches partial words of the partial query.*” *See* Section VIII.B.

171. As discussed above with respect to Claim 13, a skilled artisan would understand the combination of Sareen, Aravamudan and Anick to determine full queries using the indexes disclosed in Sareen. *See* Section IX.B.

172. Aravamudan discloses returning “Brad Pitt” as the full query for the partial words “B P,” “BR P,” and “B PI” (“*match[ing] the partial words in the same order*”). Aravamudan, [0033]. A skilled artisan would understand that this functionality, as implemented in Sareen’s system would return a full query “*that matches the partial words of the received partial query in the same order.*”

**X. GROUND 3: CLAIMS 15-18 ARE OBVIOUS IN VIEW OF SAREEN IN COMBINATION WITH KAMVAR, ARAVAMUDAN AND ANICK.**

**A. Claim 15**

**1. [15p] Preamble**

173. The preamble of Claim 15 recites “[*a*] *computer implemented method for a computer database system.*” To the extent the preamble is limiting, the combination of Sareen, Kamvar, Aravamudan and Anick renders this limitation obvious for the same reasons discussed above with respect to Claim 8. *See* Section VIII.H.1.

**2. [15a] Storing in Memory**

174. Claim 15 recites “*storing in memory data about previously received queries at the computer database system, the stored data including respective received queries and respective weight values determined based on receipt frequency of the respective received queries.*” The combination of Sareen, Kamvar, Aravamudan and Anick renders this limitation obvious.

175. Sareen discloses “*storing in memory data about previously received queries at the computer database system, the stored data including respective received queries*” for the same reasons discussed above with respect to Claim 8. See Section VIII.H.2.

176. Sareen also discloses that the stored data is assigned “*respective weight values determined based on receipt frequency of the respective received queries.*” For example, Sareen discloses that its data structure may comprise “a list of a predetermined number of most frequently requested queries beginning with the portion of the query.” Sareen, [0033]. Sareen further explains that this “list may be sorted in a descending order of frequency, so that the most common requested query beginning with the portion of the query is first listed” (*i.e.*, according to “*respective weight values determined based on receipt frequency*”). Sareen, [0033].

177. To the extent the claims require quantitative weights to be stored in memory, it would have been obvious to implement that functionality into Sareen’s system in view of at least the teachings of Kamvar.

178. Kamvar discloses storing an index of previously submitted search queries, where “each of the entries includes a term portion and a popularity value.” Kamvar, [0042]. “The popularity value 605 is a value that indicates how popular the associated term portion may be at any given moment in time. The popularity value 605 can be used to rank the predicted search queries according to current

popularity.” Kamvar, [0042]. As used in Kamvar, a skilled artisan would understand the term “popularity value” to be “*determined based on receipt frequency*.” For example, Kamvar discloses that the information stored about previously submitted queries may include “the last time the query was submitted or how frequently the query is submitted over time.” Kamvar, [0069]. Kamvar alternatively characterizes this information as “the popularity of the query or how recently in time a query was accessed.” Kamvar, [0054].

179. A skilled artisan would have been motivated to modify Sareen to disclose this functionality disclosed in Kamvar to “increase the likelihood that the user will be presented with an intended query term.” Kamvar, [0042]. Sareen’s system already presents a list of previously submitted queries in order of relative frequency. Sareen, [0033]. A skilled artisan would have recognized that storing in memory a quantitative frequency weight for each term could increase the likelihood that the system presents the user with an intended query. For example, a skilled artisan would understand that, by storing quantitative frequency weights rather than order the queries by relative weight, the system would be able to better evaluate the relevance of each matching full query.

180. A skilled artisan would have had a reasonable expectation of success in making this combination. As discussed above, Sareen’s system already evaluates the frequency of query submissions and orders the previously submitted queries

accordingly. Sareen, [0033]. A skilled artisan would have understood this process to require that some quantitative measure of frequency, such as the number of times that query has been submitted, is associated with each term. It would have been well within the ordinary abilities of a skilled artisan to store this quantitative measure of frequency in memory as disclosed by Kamvar.

### **3. [15b] Receiving a Partial Query**

181. Claim 15 recites “*receiving at a server of the computer database system a partial query submitted by a user at a remote user client system, the submitted partial query including an abbreviated or incomplete search query which is not fully representative of an entire search query desired by the user.*” The combination of Sareen, Kamvar, Aravamudan and Anick renders this limitation obvious for the same reasons discussed above with respect to Claim 8. *See* Section VIII.H.5.

### **4. [15c] Interpreting Each Word of the Partial Query as a Partial Word**

182. Claim 15 recites “*at the server, interpreting each word of the received partial query as a partial word of the entire search query desired by the user.*” The combination of Sareen, Kamvar, Aravamudan and Anick renders this limitation obvious for the same reasons discussed above with respect to Claim 13. *See* Section IX.B.

## **5. [15d] Identifying Matching Queries**

183. Claim 15 recites “*at the server, identifying in the memory previously received queries that match the partial words specified in the partial query.*” The combination of Sareen, Kamvar, Aravamudan and Anick renders this limitation obvious.

184. As discussed above with respect to Claim 13, the combination of Sareen, Kamvar, Aravamudan and Anick renders obvious “*identifying . . . previously received queries that match the partial words specified in the partial query.*” See Section IX.B.

185. As discussed above with respect to Claim 13, a skilled artisan would understand that the process of “*identifying . . . previously received queries*” involves consulting query indexes, which are stored “*in memory.*” See Sections IX.B; X.A.2.

## **6. [15e] Selecting a Full Query**

186. Claim 15 recites “*at the server, selecting as a full query a matching query which is most popular based on the weight values, the full query being better representative of the entire search query desired by the user.*” The combination of Sareen, Kamvar, Aravamudan and Anick renders this limitation obvious.

187. As discussed above with respect to Claim 2, Sareen discloses “*selecting as the full query a matching query which is most popular.*” See Section VIII.B.

188. As discussed above with respect to Claim 7, a skilled artisan would understand the “most frequently requested queries” to be the “*most popular based on the weight values.*” See Section VIII.G; *see also* Section X.A.2. Indeed, a skilled artisan would understand that the determination of the most popular matching query in the combination of Sareen, Kamvar, Aravamudan and Anick would be made based on the “*respective weight values*” stored in memory according to Kamvar’s disclosure. See Section X.A.2.

189. As discussed above with respect to Claim 8, Sareen discloses “*the full query being better representative of the entire search query desired by the user.*” See Sections VIII.H.6.

#### **B. Claim 16**

190. Claim 16 recites “[t]he method of claim 15 further comprising: *submitting the full query from the server to a search engine; receiving search results at the server from the search engine; and communicating a web page including at least some of the search results from the server to the user.*” The combination of Sareen, Kamvar, Aravamudan and Anick renders this limitation obvious for the same reasons discussed above with respect to Claim 10. See Section VIII.J.

#### **C. Claim 17**

191. Claim 17 recites “[t]he method of claim 16 further comprising: *at the server, based on the weights, determining one or more alternative queries that match*

*the submitted partial query, and wherein communicating the web page comprises communicating a web page including at least some of the one or more alternative queries.”* The combination of Sareen, Kamvar, Aravamudan and Anick renders this limitation obvious for the same reasons discussed above with respect to Claims 11 and 15. *See* Sections VIII.K, X.A.5.

**D. Claim 18**

192. Claim 18 recites “[t]he method of claim 17 wherein communicating the web page comprises communicating a web page with one or more hyperlinks corresponding to the one or more alternative queries, the hyperlinks being selectable by the user viewing the web page to select one of the one or more alternative queries.” The combination of Sareen, Kamvar, Aravamudan and Anick renders this limitation obvious.

193. As discussed above with respect to Claim 11, Anick discloses communicating a web page including at least some of the one or more alternative queries. *See* Section VIII.K. Anick further discloses displaying the alternative queries along with two “tags” (*i.e.*, “a hyperlink corresponding to the one or more alternative queries”). Anick, 8:3-5; *see also id.* at Fig. 2. Anick discloses that these tags are selectable. For example, Anick discloses that “[w]hen the user presses a tag 138, the term 136 that corresponds to the tag 138 is added to initial query 132 and the whole process repeats with the new query.” Anick, 8:10-12. Anick likewise

explains that “[w]hen the user presses another tag 139, the term 135 that corresponds to the tag 138 replaces the initial query 132 and the search engine server process that term 136 as the new query.” Anick, 8:13-16.

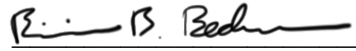
194. A skilled artisan would have been motivated to combine this functionality with Sareen’s system for the reasons discussed above with respect to claim 11. *See* Section VIII.K. A skilled artisan would have been motivated to include in the displayed webpage the selectable tags of Anick to further aid the user in quickly refining his or her search without having to enter additional text.

## **XI. CONCLUSION**

195. This declaration represents the opinions that I have formed up to date based on the material currently available to me. I reserve the right to revise, supplement, and/or amend my opinions stated in this declaration based on new information that becomes available to me and on my continuing analysis of the materials available to me.

196. I hereby declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct, and that all statements made of my own knowledge are true and correct, and that all statements made on information and belief are believed to be true. I understand that willful false statements and the like are punishable by fine or imprisonment, or both (18 U.S.C. § 1001).

Executed on April 27, 2023, at Windsor, MA.

A handwritten signature in black ink, appearing to read "B. B. Bederson", written over a horizontal line.

Benjamin B. Bederson, Ph.D.