

EXHIBIT B

**IN THE UNITED STATES DISTRICT COURT
EASTERN DISTRICT OF TEXAS
MARSHALL DIVISION**

NORTHEASTERN UNIVERSITY and
JARG CORP.,

Plaintiffs,

v.

GOOGLE INC.,

Defendant.

Civil Action No. 2:07-CV-486 (TJW)

Jury Trial Demanded

JOINT CLAIM CONSTRUCTION AND PREHEARING STATEMENT

Pursuant to Eastern District of Texas Patent Rule 4-3, Plaintiffs Northeastern University and Jarg Corporation and Defendant Google Inc. hereby provide their Joint Claim Construction and Prehearing Statement.

I. AGREED CLAIM CONSTRUCTIONS

The parties have met and conferred regarding their Rule 4-2 exchange and have agreed to the meaning of certain claim terms, phrases or clauses. These terms, and their agreed meanings, are set forth in Exhibit A (attached).

II. DISPUTED CLAIM CONSTRUCTIONS

With respect to those terms, phrases, or clauses on which the parties could not reach agreement, the parties have set forth their respective proposed constructions, including the intrinsic and extrinsic evidence relied upon in support thereof, in Exhibits B and C (attached). The parties reserve the right to rely on and rebut any evidence cited by any party.

III. ANTICIPATED LENGTH OF CLAIM CONSTRUCTION HEARING

The parties believe that it would be appropriate for the Court to allocate two hours for the claim construction hearing, with one hour allocated to each side (including reserved rebuttal time).

IV. CLAIM CONSTRUCTION LIVE WITNESSES

The parties do not anticipate calling live witnesses at the claim construction hearing.

V. CLAIM CONSTRUCTION PREHEARING CONFERENCE

The parties do not believe a claim construction prehearing conference is needed.

Dated: August 28, 2009

Respectfully submitted,

VINSON & ELKINS L.L.P.

By: /s/ Michael A. Valek

William B. Dawson (TX Bar No. 05603600)

VINSON & ELKINS L.L.P.

2001 Ross Avenue, Suite 3700

Dallas, Texas 75201-2975

Telephone: (214) 220-7926

Facsimile: (214) 999-7926

E-mail: bdawson@velaw.com

David B. Weaver (TX Bar No. 00798576)

Christopher V. Ryan (TX Bar No. 24037412)

Michael Valek (TX Bar No. 24044028)

R. Floyd Walker (TX Bar No. 24044751)

Stephen C. Stout (TX Bar No. 24060672)

Meredith J. Fitzpatrick (TX Bar No. 24059753)

VINSON & ELKINS L.L.P.

2801 Via Fortuna, Suite 100

Austin, Texas 78746

Telephone: (512) 542-8400

Facsimile: (512) 236-3338

E-mail: dweaver@velaw.com

cryan@velaw.com

mvalek@velaw.com

fwalker@velaw.com

sstout@velaw.com

mfitzpatrick@velaw.com

Otis W. Carroll, Jr. (TX Bar No. 03895700)

Collin Maloney (TX Bar No. 00794219)

IRELAND CARROLL & KELLEY

6101 S. Broadway, Suite 500

Tyler, TX 75703

Telephone: (903) 561-1600

Facsimile: (903) 581-1071

Attorneys for Plaintiffs

NORTHEASTERN UNIVERSITY AND JARG
CORPORATION

Dated: August 28, 2009

FISH & RICHARDSON P.C.

By: /s/ Shelley K. Mack

Michael E. Jones (SBN 10929400)

mikejones@potterminton.com

POTTER MINTON

A Professional Corporation

110 N. College, Suite 500

Tyler, TX 75702

Telephone: (903) 597-8311

Facsimile: (903) 593-0846

Ruffin B. Cordell (SBN 04820550)

cordell@fr.com

FISH & RICHARDSON P.C.

1425 K Street, N.W., 11th Floor

Washington, DC 20005-3500

Telephone: (202) 783-5070

Facsimile: (202) 783-2331

Jason W. Wolff (CA SBN 215819)

wolff@fr.com

FISH & RICHARDSON P.C.

12390 El Camino Real

San Diego, CA 92130

Telephone: (858) 678-5070

Facsimile: (858) 678-5099

Howard G. Pollack (*Admitted Pro Hac Vice*)

pollack@fr.com

Shelley K. Mack (*Admitted Pro Hac Vice*)

mack@fr.com

Jerry T. Yen (CA SBN 247988)

yen@fr.com

FISH & RICHARDSON P.C.

500 Arguello Street, Suite 500

Redwood City, CA 94063

Telephone: (650) 839-5070

Facsimile: (650) 839-5071

Attorneys for Defendant

GOOGLE INC.

CERTIFICATE OF SERVICE

The undersigned certifies that the foregoing document was filed electronically in compliance with Local Rule CV-5(a). As such, this document was served on all counsel who are deemed to have consented to electronic service. Local Rule CV-5(a)(3)(A). Pursuant to Fed. R. Civ. P. 5(d) and Local Rule CV-5(d) and (e), all other counsel of record not deemed to have consented to electronic service were served with a true and correct copy of the foregoing by email and/or fax, on this the 18th day of December, 2008.

/s/ Michael A. Valek

Michael A. Valek

EXHIBIT A

PARTIES' AGREED CONSTRUCTIONS

No.	Term	Claim(s)	Agreed Construction
1.	fuzzy queries	claim 1	imprecise or inexact requests for information from a database, the result of which does not necessarily contain each term in the query

EXHIBIT B

**PLAINTIFFS' PROPOSED CONSTRUCTIONS FOR
DISPUTED CLAIM TERMS AND EVIDENTIARY SUPPORT**

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
1.	non-relational, distributed database system	claims 1, 8, 13	<p><u>Construction:</u></p> <p>a database not using a relational model that is distributed among a plurality of interconnected computer nodes</p> <p><u>Intrinsic Support:</u></p> <p>Abstract; Fig. 1; 1:5-6; 1:10-62; 1:65-2:2; 2:3-18; 2:56-3:16; 3:17-3:36; 3:46-50; 4:15-21; 4:22-55; 7:50-65; 10:4-17; 10:18-20</p> <p>June 7, 1996 Amendment p. 7-19; December 4, 1996 Amendment p. 7-9; May 14, 1997 Amendment p. 3-4;</p>
2.	<p>a plurality of home nodes and a plurality of query nodes connected by a network</p> <p>a plurality of home nodes; and a plurality of query nodes; said plurality of home nodes and said plurality of query nodes connected by a network</p>	<p>claim 1</p> <p>claims 8, 13</p>	<p><u>Construction:</u></p> <p>The claim language has its plain and ordinary meaning; no further construction necessary.</p> <p><u>Intrinsic Support:</u></p> <p>Fig. 1-2; Abstract; 1:65-2:18; 2:66-3:36; 3:51-59; 4:8-14; 4:22-35; 7:23-49; 10:4-20.</p> <p>June 7, 1996 Amendment p. 16; May 14, 1997 Amendment p. 2-3;</p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
3.	randomly selecting	claim 1	<p><u>Construction:</u></p> <p>selecting without an apparent pattern</p> <p><u>Intrinsic Support:</u></p> <p>Fig. 1; June 7, 1996 Amendment p. 13, 16.</p> <p>“High-Performance, Distributed Information Retrieval,” Kenneth Baclawski and J. Elliott Smith. p. 3.</p> <p>“A distributed approach to high-performance information retrieval,” Kenneth Baclawski and J. Elliott Smith, p. 4</p> <p>"A unified approach to high-performance, vector-based information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 4, 9, 13.</p> <p>"KEYNET: An architecture and protocol for high-performance semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 14, 19-20.</p> <p>"An abstract model for semantically rich information retrieval," Kenneth Baclawski and Dan A. Simovici, p. 4.</p> <p><u>Extrinsic Support:</u></p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
			<p>“random adj 1 : lacking or seeming to lack a regular plan, purpose or pattern” WEBSTER’S THIRD NEW INTERNATIONAL DICTIONARY OF THE ENGLISH LANGUAGE 1880 (Phillip Babcock Gove ed., 1993).</p>
4.	query fragment	claims 1, 8, 13	<p><u>Construction:</u></p> <p>a sub-part or piece of a query</p> <p><u>Intrinsic Support:</u></p> <p>Figs. 4, 7, 8; Abstract; 2:4-17; 3:25-50; 4:60-66; 7:23-49; 8:66-9:5; 9:30-53; 9:61-10:3.</p> <p>May 4, 1997 Amendment at p. 2-3.</p> <p>Dec. 4, 1996 Amendment at p. 7.</p> <p>June 7, 1996 Amendment p. 7-17.</p> <p>“High-Performance, Distributed Information Retrieval,” Kenneth Baclawski and J. Elliott Smith. p. 2.</p> <p>“A distributed approach to high-performance information retrieval,” Kenneth Baclawski and J. Elliott Smith, p. 8-9</p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
			<p>"A unified approach to high-performance, vector-based information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 4, 8-9, 14.</p> <p>"KEYNET: An architecture and protocol for high-performance semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 14, 19-21.</p> <p>"An abstract model for semantically rich information retrieval," Kenneth Baclawski and Dan A. Simovici, p. 17-26.</p> <p>"KEYNET: Fast indexing for semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 10-15.</p> <p>"High-Performance Indexing and Retrieval for Object-Oriented Databases," Kenneth Baclawski, p. 7, 10, 19</p> <p><u>Extrinsic Support:</u></p> <p>“fragment n 1 : a part broken off : a small detached portion : an imperfect or incomplete part” WEBSTER’S THIRD NEW INTERNATIONAL DICTIONARY OF THE ENGLISH LANGUAGE 901 (Phillip Babcock Gove ed., 1993)</p>
5.	hashing / hashes	claims 1, 8, 13	<p><u>Construction:</u></p> <p>a computer technique whereby one or more functions are used to transform values into corresponding values</p> <p><u>Intrinsic Support:</u></p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
			<p>Figs. 4, 7a, 8; Abstract; 2:3-11; 3:25-50; 3:60-62; 5:2-5; 6:65-67; 7:23-38.</p> <p>June 7, 1996 Amendment, p. 7, 14-16.</p> <p>Dec. 4, 1996 Amendment, p. 7.</p> <p>May 14, 1997 Amendment, p. 4.</p> <p>“High-Performance, Distributed Information Retrieval,” Kenneth Baclawski and J. Elliott Smith. p. 3.</p> <p>“A distributed approach to high-performance information retrieval,” Kenneth Baclawski and J. Elliott Smith, p. 4-5</p> <p>"A unified approach to high-performance, vector-based information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 10, 17</p> <p>"KEYNET: An architecture and protocol for high-performance semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 6., 14-15, 21</p> <p>"An abstract model for semantically rich information retrieval," Kenneth Baclawski and Dan A. Simovici, p. 5.</p> <p>"KEYNET: Fast indexing for semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 3, 8-15.</p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
			<p><u>Extrinsic Support:</u></p> <p>“Hashing (or hash coding) is a word coined by computer programmers to describe a general class of operations done to transform one or more fields (usually a key) into a different (usually more compact) arrangement. Probably, “hashing” was first coined because it seemed that “hash” was being made out of integral pieces of data. The rationale for hashing is developed more fully in the article on table lookup, dealing with key transformation to convert naturally occurring, diverse, ill-structured, scattered key fields into compact, easily manipulated fields—usually some numeric, computer oriented field such as a word or double word, or a computer memory address to facilitate subsequent references. The transformation from the natural field to the hash address is only a one-way process, however; the natural field cannot be decoded or reconstructed from the hash. Also, the hashed field may not represent only one unique natural field; many natural fields could hash into the same value.” ENCYCLOPEDIA OF COMPUTER SCIENCE AND ENGINEERING 681 (Anthony Ralston & Edwin D. Reilly, Jr. eds.,1983).</p> <p>“Hashing (1) (* Processing) A technique for converting the values within fields into more ‘compact’ representations of the same values. Usually, the values being converted are keys. (2) (* Addressing) An addressing technique used to store and retrieve data in a file. Hashing uses keys in an index to determine the location of the data in the file.” ROBERT A EDMUNDS, THE PRENTICE-HALL STANDARD GLOSSARY OF COMPUTER TERMINOLOGY, 191-192 (1985).</p> <p>“Hashing: Hashing refers to a storage mechanism where data items are stored at locations that are determined by a mathematical function of the data. For example, suppose you need to store a list of 100 numbers in memory locations whose addresses run from 1 to 100. One example of a hashing</p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
			<p>function is to calculate the remainder when the number is divided by 100 and then store it at that location. For example, the number 538 would be stored at memory location 38. The use of hashing makes it possible to store and retrieve the data items quickly since it is not necessary to search through the list in order to find the item. However, there is one complication: A hashing function will sometimes assign more than one data item to the same address. For example, using the rule given above, the number 638 would also be stored in location 38. To avoid that problem, a hashing system needs to be able to resolve collisions by storing the new data item in a separate place.” DOUGLAS DOWNING & MICHAEL COVINGTON, BARRON’S DICTIONARY OF COMPUTER TERMS 158 (3d ed. 1992).</p> <p>“Hashing: (1) Key-to-address transformation in which the keys determine the location of the data. (2) The process of applying a formula to a record key to yield a number that represents a disk address.” DONALD D. SPENCER, COMPUTER DICTIONARY (3d ed. 1992).</p> <p>“Hashing: In database management, an indexing technique in which the value of a key (record identifier) is numerically manipulated to directly calculate either the location of its associated record in a file or the starting point for a search for the associated record.” MICROSOFT COMPUTER DICTIONARY 193 (2d ed. 1994).</p>
6.	a first portion and a second portion	claims 1, 8, 13	<p><u>Construction:</u></p> <p>The claim language has its plain and ordinary meaning; no further construction necessary.</p> <p><u>Intrinsic Support:</u></p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
			3:36-50; 4:64-5:2; 6:59-65; 7:23-38; 7:50-55.
7.	<p>transmitting, by said selected home node, each said hashed query fragment of said plurality of query fragments to a respective one of said plurality of query nodes indicated by said first portion of each said hashed query fragment</p> <p>transmits each said hashed query fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed query fragment</p> <p>transmitting a query message containing every said hashed query fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed query fragment</p>	<p>claim 1</p> <p>claim 8</p> <p>claim 13</p>	<p><u>Construction:</u></p> <p>The claim language has its plain and ordinary meaning; no further construction necessary.</p> <p><u>Intrinsic Support:</u></p> <p>Figs. 1, 2, 3, 8, 8a; Abstract; 2:3-18; 3:17-50; 4:23-36; 7:23-49.</p> <p>"KEYNET: An architecture and protocol for high-performance semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 3.</p> <p>"KEYNET: Fast indexing for semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 9, 13.</p> <p>June 7, 1996 Amendment, p. 14.</p>
8.	using, by said query node,	claim 1	<u>Construction:</u>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
	<p>said second portion of said respective hashed query fragment to access data according to a local hash table located on said query node</p> <p>each said query node uses said second portion of said hashed query fragment to access data according to a local hash table located on said query node</p> <p>said query node, upon receipt of said query message, using said second portion of said hashed query fragment to access data according to a local hash table located on said query node</p>	<p>claim 8</p> <p>claim 13</p>	<p>The claim language has its plain and ordinary meaning; no further construction necessary.</p> <p><u>Intrinsic Support:</u></p> <p>Figs. 1, 2, 3a; 2:12-18; 3:37-59; 6:1-28; 7:50-65</p> <p>"A distributed approach to high-performance information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 4-5.</p> <p>"A unified approach to high-performance, vector-based information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 10.</p> <p>"KEYNET: An architecture and protocol for high-performance semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 14-15.</p> <p>"An abstract model for semantically rich information retrieval," Kenneth Baclawski and Dan A. Simovici, p. 5.</p> <p>June 7, 1996 Amendment, p. 14.</p>
9.	local hash table	claims 1, 8, 13	<p><u>Construction:</u></p> <p>a table that associates hash values with other data</p> <p><u>Intrinsic Support:</u></p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
			<p>Figs. 1, 2, 3a; 2:12-18; 3:37-59; 6:1-28; 7:50-65; 8:64-66;</p> <p>"A distributed approach to high-performance information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 2, 4, 5</p> <p>"A unified approach to high-performance, vector-based information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 2, 3, 10.</p> <p>"KEYNET: An architecture and protocol for high-performance semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 10, 12, 14-15.</p> <p>"An abstract model for semantically rich information retrieval," Kenneth Baclawski and Dan A. Simovici, p. 3, 5.</p> <p>"KEYNET: Fast indexing for semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 6, 12, 17.</p> <p>May 14, 1997 Amendment, p. 4.</p> <p>June 7, 1996 Amendment, p. 14.</p> <p><u>Extrinsic Support:</u></p> <p>"Hash table: A table of information that is accessed by way of a shortened search key (the hash value). Using a hash table minimizes average search time." IBM DICTIONARY OF COMPUTING 309 (George McDaniel ed., 10th ed. 1994).</p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
			<p>“Hash Table: A data structure that maps a set of keys to slots from a hashing function.” Ricardo A. Baeza-Yates, Introduction to Data Structures and Algorithms Related to Information Retrieval, in INFORMATION RETRIEVAL: DATA STRUCTURES AND ALGORITHMS 13, 20 (William B. Frakes & Ricardo A. Baeza-Yates, eds., 1992).</p>
10.	<p>returning, by each said query node</p> <p>each said query node ... returns</p> <p>said query node ... returning</p>	<p>claim 1</p> <p>claim 8</p> <p>claim 13</p>	<p><u>Construction:</u></p> <p>The claim language has its plain and ordinary meaning; no further construction necessary.</p> <p><u>Intrinsic Support:</u></p> <p>Figs. 1, 2, 3a; 2:12-18; 3:55-65; 4:2-7; 7:50-65; 8:8-15</p> <p>June 7, 1996 Amendment, p. 18.</p>
11.	<p>predetermined degree of relevance</p>	<p>claims 3, 9</p>	<p><u>Construction:</u></p> <p>a degree of relevance that is determined before returning accessed data to the user</p> <p><u>Intrinsic Support:</u></p> <p>3:60-4:21; 9:44-47.</p> <p>"High Performance, Distributed Information Retrieval," by Kenneth Baclawski and J. Elliott Smith, p. 2.</p>

No.	Term	Claim(s)	Plaintiffs' Proposed Construction & Evidentiary Support
			<p>"High-Performance Indexing and Retrieval for Object-Oriented Databases," Kenneth Baclawski, p. 24.</p> <p>"A distributed approach to high-performance information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 5, 7, 9.</p> <p>"A unified approach to high-performance, vector-based information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 9, 10.</p> <p>"KEYNET: An architecture and protocol for high-performance semantically rich information retrieval," Kenneth Baclawski* and J. Elliott Smith, p. 4, 6-8, 10, 15, 20.</p> <p>"An abstract model for semantically rich information retrieval," Kenneth Baclawski and Dan A. Simovici, p. 1, 5, 14.</p> <p>"KEYNET: Fast indexing for semantically rich information retrieval," Kenneth Baclawski and J. Elliott Smith, p. 2, 4-7, 9, 11, 14.</p> <p>June 7, 1996 Amendment, p. 16.</p> <p>December 4, 1996 Amendment at p. 8-9.</p>

EXHIBIT C

GOOGLE, INC.'S PROPOSED CONSTRUCTIONS OF
DISPUTED CLAIM TERMS

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
1.	non-relational, distributed database system	claims 1, 8, 13	<p>a database, stored across multiple computers on a network, wherein data objects exist independently of their attribute values, and wherein data is not extracted using relational algebra</p> <p><u>Intrinsic Support:</u></p> <p>'593 patent at Abstract; col. 1:10-37; col. 1:65-2:18; col. 2:66-3:59; col. 4:23-7:38; col. 10:25-51; col. 11:24-44; col. 12:6-33; Fig. 1, Fig. 2, Fig. 8, and Fig. 8a.</p> <p>'593 Prosecution History, June 7, 1996 Amendment at 7-14, 16-19; Dec. 11, 1996 Amendment at 8-9; May 14, 1997 Amendment at 3.</p> <p>Chaturvedi et al., <i>Scheduling the Allocation of Data Fragments in a Distributed Database Environment: A Machine Learning Approach</i>, IEEE TRANS ON ENG'G MGMT., vol. 41, no. 2, 1994, pp. 194-206. (GN 524 – 537)</p> <p>Houtsma et al., <i>Parallel Hierarchical Evaluation for Transitive Closure Queries</i>, IEEE Apr. 1991. (GN 4932 – 4941)</p> <p>U.S. Patent 4,811,199. (GN 7147 – 7162)</p> <p><u>Extrinsic Support:</u></p> <p>“Relational database: A database in which the data are organized and accessed according to relations.” <i>Dictionary of Computing</i>. Research Triangle Park, NC: International Business Machines Corporation, 1991, p. 475. (GN 292968)</p> <p>“Relational database: A database is which data are organized into one or more relations that may be manipulated using a relational algebra.” Christopher Booth, ed. <i>The New IEEE Standard Dictionary of Electrical and Electronics Terms</i> (5th Ed.: Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 1106.</p>

Google, Inc.'s Proposed Construction & Evidentiary Support		
No.	Term	Claim(s)
		<p>(GN 300219)</p> <p>“Relational database: A database is which data are organized into one or more relations that may be manipulated using a relational algebra.” <i>IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries</i> (Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 170. (GN 300234)</p> <p>“Relation: (1) In a relational database, a set of entity occurrences that have the same attributes ... (3) In a relational database, a table that identifies entities and their attributes. Synonymous with flat file” <i>Dictionary of Computing</i>. Research Triangle Park, NC: International Business Machines Corporation, 1991, p. 475. (GN 292968)</p> <p>“Relation: In a relational data model or relational database, a set of tuples, each of which has the same attributes.” Christopher Booth, ed. <i>The New IEEE Standard Dictionary of Electrical and Electronics Terms</i> (5th Ed.: Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 1106. (GN 300219)</p> <p>“Relation: In a relational data model or relational database, a set of tuples, each of which has the same attributes.” <i>IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries</i> (Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 170. (GN 300234)</p> <p>“Relational database management system: A database organization scheme that treats files as tables of data in which the rows represent fixed-length records and columns represent fields. Multiple keys can be used for retrieving the data stored within the database. A database in which some data items in one type of record refer to records of a different type. Relational databases give the user the flexibility to link (join, or create a relationship between) information stored in many disk files. It allows users to interchange and cross-reference information between two different types of records, such as comparing the</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>information in a group of invoices to the information in an inventory. Most people do not need relational databases. To merely keep track of a mailing list doesn't require a relational database, nor is such a database needed to keep a simple inventory of something. However, those who want to print out a mailing list of people who ordered products from their inventory will need a relational database. Relational databases are more powerful, more complex, more difficult to use, and more expensive than other database systems.”</p> <p><i>Webster's New World Dictionary of Computer Terms</i> (4TH ed.) (Prentice Hall: New York, NY), 1992, p. 353.</p> <p>“Relational database management system (RDBMS): A database management system based on the relational model. This claim is often made (particularly for personal computer packages) principally on the grounds that the data is treated as a series of two-dimensional tables, known as relations. Stricter criteria would require also that algebraic operations, such as JOIN or PROJECT, could be used to manipulate the data and to create new tables based on various combinations of the original tables.” Gunton, Tony. <i>A Dictionary of Information Technology and Computer Science</i>, 257, 2nd ed. Oxford, UK: NCC Blackwell Ltd. 1993, p. 257. (GN 292982)</p> <p>Kenneth Baclawski and J. Elliott Smith, <i>A Unified Approach to High-Performance, Vector-Based Information Retrieval</i>, March 21, 1994. (JAR 294–313)</p>
2.	<p>a plurality of home nodes and a plurality of query nodes connected by a network</p> <p>a plurality of home nodes; and a plurality of query nodes; said plurality of</p>	<p>claim 1</p> <p>claims 8, 13</p>	<p>a plurality of home nodes and query nodes connected by a network arranged with no central server and wherein, for any given query, any node may be defined as a home node or a query node</p> <p><u>Intrinsic Support:</u></p> <p>'593 patent at Abstract; col. 1:65-2:18; col. 2:66-3:59; col. 4:8-7:38; col. 10:25-51; col. 11:24-44; col. 12:6-33; Fig. 1, Fig. 2, Fig. 8, Fig. 8a.</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
	home nodes and said plurality of query nodes connected by a network		<p>'593 Prosecution History, April 16, 1996 Office Action at 2-3; June 7, 1996 Amendment at 13, 16; May 14, 1997 Amendment at 2-3.</p> <p>Chaturvedi et al., <i>Scheduling the Allocation of Data Fragments in a Distributed Database Environment: A Machine Learning Approach</i>, IEEE TRANS ON ENG'G MGMT., vol. 41, no. 2, 1994, pp. 194-206. (GN 524 – 537)</p> <p>Houtsma et al., <i>Parallel Hierarchical Evaluation for Transitive Closure Queries</i>, IEEE Apr. 1991. (GN 4932 – 4941)</p> <p>U.S. Patent 4,811,199. (GN 7147 – 7162)</p> <p><u>Extrinsic Support:</u> Kenneth Baclawski and J. Elliott Smith, <i>A Unified Approach to High-Performance, Vector-Based Information Retrieval</i>, March 21, 1994. (JAR 294–313)</p>
3.	randomly selecting	claim 1	<p>selecting by chance, independently of preceding selections, where each item in the set has equal probability of being chosen</p> <p><u>Intrinsic Support:</u> '593 patent at col. 3:17-25; col. 10:25-51; col. 11:24-44; col. 12:6-33; Fig. 1, Fig. 2.</p> <p>'593 Prosecution History, June 7, 1996 Amendment at p. 13, 16; September 11, 1996 Office Action at 2; March 13, 1997 Office Action at 2; May 14, 1997 Response to Office Action at 2.</p> <p>Chaturvedi et al., <i>Scheduling the Allocation of Data Fragments in a Distributed Database Environment: A Machine Learning Approach</i>, IEEE TRANS ON ENG'G MGMT., vol. 41, no. 2, 1994, pp. 194-206. (GN 524 – 537)</p>

Google, Inc.'s Proposed Construction & Evidentiary Support		
No.	Term	Claim(s)
		<p>Houtsma et al., <i>Parallel Hierarchical Evaluation for Transitive Closure Queries</i>, IEEE Apr. 1991. (GN 4932 – 4941)</p> <p><u>Extrinsic Support:</u></p> <p>“Random: 1. occurring or done without definite aim, reason, or pattern: random examples. 2. Statistics. of or characterizing a process of selection in which each item of a set has an equal probability of being chosen. . . . 5. at random, without regard to rules, schedules, etc.; haphazardly.” <i>Random House Webster’s College Dictionary</i>, New York, NY: Random House Inc. 1991, p. 1116. (GN 300206)</p> <p>“Random: . . . b. Statistics. Governed by or involving equal chances for each of the actual or hypothetical members of a population; also, produced or obtained by a random process (and therefore completely unpredictable in detail) the movement of something in successive steps . . . each step being governed by chance independently of preceding steps.” J.A. Simpson & E.S.C. Wiener, eds. <i>Oxford English Dictionary</i>, 2d ed., vol. 13, 1989 (Clarendon Press: Oxford, UK), p. 168. (GN 300405)</p> <p>“Random: (3) (modeling and simulation). Pertaining to a process or variable whose outcome or value depends on chance or on a process that simulates chance, often with the implication that all possible outcomes or values have an equal probability of occurrence; for example, the outcome of flipping a coin or executing a computer-programmed random number generator.” Christopher Booth, ed. <i>The New IEEE Standard Dictionary of Electrical and Electronics Terms</i> (5th Ed.: Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 1064. (GN 300218)</p> <p>“Random: Pertaining to a process or variable whose outcome or value depends on chance or on a process that simulates chance, often with the implication that all possible outcomes or values have an equal probability of occurrence; for</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>example, the outcome of flipping a coin or executing a computer-programmed random number generator.” <i>IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries</i> (Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 167. (GN 300233)</p> <p>Kenneth Baclawski and J. Elliott Smith, <i>A Unified Approach to High-Performance, Vector-Based Information Retrieval</i>, March 21, 1994. (JAR 294–313)</p>
4.	query fragment	claims 1, 8, 13	<p>a part of a query consisting of a limited number of attributes and attribute values joined by relationships, specified in the same formal, artificial language and ontology which describes the attribute values of objects of the database</p> <p><u>Intrinsic Support:</u></p> <p>'593 patent at Abstract; col. 1:10-31; col. 2:3-18; col. 3:25-4:7; col. 4:22-36; col. 4:60-5:29; col. 10:25-51; col. 11:24-44; col. 12:6-33; Fig. 2, Fig. 8, Fig. 8a.</p> <p>'593 Prosecution History, June 7, 1996 Amendment at 9-16; Dec. 11, 1996 Amendment at 8-9; May 14, 1997 Response to Office Action at 2-3.</p> <p>Chaturvedi et al., <i>Scheduling the Allocation of Data Fragments in a Distributed Database Environment: A Machine Learning Approach</i>, IEEE TRANS ON ENG'G MGMT., vol. 41, no. 2, 1994, pp. 194-206. (GN 524 – 537)</p> <p>Houtsma et al., <i>Parallel Hierarchical Evaluation for Transitive Closure Queries</i>, IEEE Apr. 1991. (GN 4932 – 4941)</p> <p>U.S. Patent 4,811,199. (GN 7147 – 7162)</p> <p><u>Extrinsic Support:</u></p> <p>“The '593 query fragments are defined at '593 column 1, lines 27-31, and consist of a part of the query with a limited number of attribute values joined by</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>relationships.” U.S. Patent No. 6,505,191 Prosecution History, July 3, 2002 Response to office action at 5. (GN 299941 – 300172)</p> <p>Kenneth Baclawski and J. Elliott Smith, <i>A Unified Approach to High-Performance, Vector-Based Information Retrieval</i>, March 21, 1994. (JAR 294–313)</p> <p>Kenneth Baclawski and J. Elliott Smith, <i>KEYNET: Fast Indexing for Semantically Rich Information Retrieval</i>, December 7, 1993 (JAR 828–845)</p> <p>“Prototype Specifications . . . The hash algorithm is taken from Knuth, volume 3, section 6.4.” (JAR 219469)</p>
5.	hashing / hashes	claims 1, 8, 13	<p>performing a mathematical function on a key value to generate the address of the location of data associated with the key value</p> <p><u>Intrinsic Support:</u></p> <p>'593 patent at Abstract; col. 1:33-62; col. 2:3-12; col. 3:25-4:7; col. 4:60-5:29; col. 6:39-7:38; col. 8:61-9:2; col. 10:25-51; col. 11:24-44; col. 12:6-33; Fig. 8, Fig. 8a.</p> <p>'593 Prosecution History, June 7, 1996 Amendment at 12, 14-16; May 14, 1997 Amendment at 3-4.</p> <p>Chaturvedi et al., <i>Scheduling the Allocation of Data Fragments in a Distributed Database Environment: A Machine Learning Approach</i>, IEEE TRANS ON ENG'G MGMT., vol. 41, no. 2, 1994, pp. 194-206. (GN 524 – 537)</p> <p>Houtsma et al., <i>Parallel Hierarchical Evaluation for Transitive Closure Queries</i>, IEEE Apr. 1991. (GN 4932 – 4941)</p> <p>U.S. Patent 4,811,199. (GN 7147 – 7162)</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p><u>Extrinsic Support:</u></p> <p>“hashing: (1) a key-to-address transformation in which the keys determine the location of the data. (2) The process of applying a formula to a record key to yield a number that represents a disk address.” <i>Webster’s New World Dictionary of Computer Terms</i> (4TH ed.) (Prentice Hall: New York, NY), 1992, p. 187. (GN 300244)</p> <p>“hashing: A technique for arranging a set of items, in which a hash function is applied to the key of each item to determine its hash value. The hash value identifies each item’s primary position in a hash table, and if this position is already occupied, the item is inserted wither in an overflow table or in another available positioning the table.” Christopher Booth, ed. <i>The New IEEE Standard Dictionary of Electrical and Electronics Terms</i> (5th Ed.: Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 586. (GN 300215)</p> <p>“hashing: A technique for arranging a set of items, in which a hash function is applied to the key of each item to determine its hash value. The hash value identifies each item’s primary position in a hash table, and if this position is already occupied, the item is inserted wither in an overflow table or in another available positioning the table.” <i>IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries</i> (Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 99. (GN 300228)</p> <p>“hashing algorithm: An algorithm used to derive an address within a specified range from a key value. A hashing algorithm is used with a random file to determine the address of the block in which a given record should be stored.” Gunton, Tony, <i>A Dictionary of Information Technology and Computer Science</i> (2nd ed.) (Oxford, UK: NCC Blackwell Ltd.), 1993, p. 136. (GN 292980)</p> <p>“Section 6.3 treats digital searching, and Section 6.4 discusses an important</p>

Google, Inc.'s Proposed Construction & Evidentiary Support		
No.	Term	Claim(s)
		<p>class of methods called hashing techniques, based on arithmetic transformations of the actual keys.” Donald Knuth, <i>The Art of Computer Programming, Volume 3, Sorting and Searching</i>, Addison-Wesley, 1973, p. 390.</p> <p>“6.4: Hashing. So far we have considered search methods based on comparing the given argument K to the keys in the table, or using its digits to govern a branching process. A third possibility is to avoid all this rummaging around by doing some arithmetical calculation on K, computing a function f(K) which is the location of K and the associated data on the table. . . . These considerations lead to a popular class of search methods commonly known as hashing or scatter storage techniques. The verb “to hash” means to chop something up or to make a mess out of it; the idea in hashing is to chop off some aspects of the key and to use this partial information as the basis for searching. We compute a hash function h(K) and use this value as the address where the search begins.” Donald Knuth, <i>The Art of Computer Programming, Volume 3, Sorting and Searching</i>, Addison-Wesley, 1973, p. 506-07.</p> <p>“Prototype Specifications . . . The hash algorithm is taken from Knuth, volume 3, section 6.4.” (JAR 219469)</p> <p>“hash: An associative access technique that maps the key via a hash function uniformly among a partitioned set of hash buckets. A key search consists of hashing the key to a bucket address and then examining the small hash bucket for records with the desired key.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 120; <i>see also generally</i> pp. 831-851.</p> <p>“Associative access on a single relation can be supported by two types of access paths that use fundamentally different approaches to solving the problem of translating attribute values into tuple addresses. These approaches, which yield fundamentally different functionality, are generally referred to as hashing</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>(key transformation) and key comparison.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 833.</p> <p>“Hashing is based on the idea of using the primary key value as a parameter to a function which returns the storage location of the tuple as a result. This is the very principle of hashing, and it has a number of interesting properties. If the transformation function can be kept simple—that is, if it does not need large data structures with search paths and routing data—then the access cost to retrieve a tuple via its key is minimal: take the key value, do some arithmetic, and find the tuple at the address delivered by the function. Ideal hash algorithms allow for one-access retrieval, which is to say that only the page holding the tuple needs to be read when accessing via the primary key. The functional principle of simple hashing is to relate to key value of a tuple and the page number in which it is stored, through a predefined function; because of this, simple hashing is not just an access method, but also a file organization technique. (This was explained in Chapter 14.) As such, it needs page address spaces with special properties, as with become obvious during the detailed description. Hash-based access paths support only queries of the type (key-attribute = const). They do not efficiently support range predicates such as key-attribute between A and B. The attribute used for hashing typically is a primary key of the relation, but can also be used for nonunique attributes.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 833.</p> <p>“Key comparison comprises all methods for maintaining a dynamic search structure on the set of values in the key attribute. These values (or compressed versions of them) can be organized into tables, lists, trees, and so on, depending on the amount of data, the type of search operations to be supported, and the storage size that is available for maintaining the search structure. If such a technique is used for a primary access path, the entire tuple can be stored in</p>

Google, Inc.'s Proposed Construction & Evidentiary Support		
No.	Term	Claim(s)
		<p>search structure (see Section 14.4.5, on key-sequenced files, in Chapter 14); otherwise, it will contain pointers to the tuples (e.g., TIDs). Sorting a file along some attribute and keeping it sorted under updates is a very simple example of a search structure based on key comparison—in this case, it is a sequential sorted list of tuples. If records are clustered within blocks according to the key attribute values, searching and scanning in sorted order can be performed efficiently.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 833.</p> <p>“Algorithms to implement associative access come in two flavors. The first one, hashing, supports access based on value equality only; it uses a transformation function that turns the attribute value into a page address where the tuple can be found.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 835.</p> <p>“As in hash-based memory organizations, there has to be a function to transform the attribute value into a file address, where the tuple will (most likely) be found.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 835.</p> <p>“[A] good hash function H has to map primary key values, which are very unevenly distributed over a large value range, into a tuple address space that is much smaller (proportional to the number of existing tuples), such that the resulting addresses are evenly distributed over the address range.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 839.</p> <p>“[i]f the hash function $h(p)$ is properly chosen, then the hash values will be approximately uniformly distributed over their range.” (JAR 108483)</p> <p>Kenneth Baclawski and J. Elliott Smith, <i>A Unified Approach to High-</i></p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p><i>Performance, Vector-Based Information Retrieval</i>, March 21, 1994 (JAR 294–313)</p> <p>Kenneth Baclawski and J. Elliott Smith, <i>KEYNET: Fast Indexing for Semantically Rich Information Retrieval</i>, December 7, 1993 (JAR 828–845)</p> <p>JAR0146229-146266</p> <p>“The examination node 102 then encodes each feature fragment of the object by using a predefined hashing function. Data in the system was previously stored locally on the various index nodes using this hashing function to generate an index to the data in the local database. Thus, the use of the same hashing function to generate an index for data storage and to generate hashed feature fragments for an information object assures that (1) data is distributed uniformly over the index nodes of the routing search engine during the storing of data and (2) the feature fragments are scattered uniformly over the index nodes during the processing of an object.” U.S. Patent No. 6,192,364, col. 6:46-58. (GN 299812 – 831)</p> <p>“The home node 107 then encodes each feature of the query by using a predefined hashing function. Data in the system was previously stored locally on the various query nodes 109 using this hashing function to generate an index to the data in the local database. Thus, the use of the same hashing function to generate an index for data storage and to generate hashed probes for a data query assures that (1) data is distributed uniformly over the query nodes 109 of the search engine during the storing of data and (2) the probes are scattered uniformly over the query nodes 109 during the processing of a query.” U.S. Patent No. 6,424,973, col. 7:58-8:2. (GN299832 – 854)</p> <p>“The home node encodes each fragment of the query by using a predefined hashing function. The same hashing function preferably is also used in</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>generating indexes to storage locations for storing data locally in local databases on the various query nodes. The use of the same hashing function to generate an index for data storage and to generate hashed probes for a query assures that data is distributed uniformly over the query nodes of the search engine during the storing of data, and that the probes are scattered uniformly over the query nodes during the processing of a query.” U.S. Patent No. 6,463,433, col. 11:18-29. (GN 299855 – 877)</p> <p>“Thus, the use of the same hashing function to generate an index for data storage and to generate hashed probes for an object assures that data is distributed uniformly over the index nodes 106 of the data warehouse during the storing of data.” U.S. Patent No. 6,470,333 Col. 7:36-41.</p> <p>“The home node 105 encodes each fragment of the query by using a predefined hashing function. Data in the distributed computer database system was previously stored locally on the various index nodes 112 using this hashing function to generate an index to the data in the local database. In particular, if a fragment includes a link then it is hashed and stored as a link fragment, while if a fragment does not include a link then it is hashed and stored as an index fragment. Thus, the use of the same hashing function to generate an index for data storage and to generate hashed probes for a query assures that 1. data is distributed uniformly over the index nodes of the search engine during the storing of data and 2. the probes are scattered uniformly over the index nodes during the processing of a query.” U.S. Patent No. 6,505,191, col. 8:61-9:8. (GN 299917 – 940)</p> <p>“The examination node 102 then encodes each feature fragment of the object by using a predefined hashing function. Data in the system was previously stored locally on the various index nodes using this hashing function to generate an index to the data in the local database. Thus, the use of the same hashing function to generate an index for data storage and to generate hashed feature</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>fragments for an information object assures that (1) data is distributed uniformly over the index nodes of the routing search engine during the storing of data and (2) the feature fragments are scattered uniformly over the index nodes during the processing of an object.” U.S. Patent No. 6,535,881, col. 6:59-7:3. (GN 299898 – 916)</p> <p>U.S. Patent No. 6,505,191 Prosecution History, July 3, 2002 Response to Office Action at 5. (GN 299941 – 300172)</p> <p><i>See generally</i> Gerald Salton, <i>Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer</i>, Addison-Wesley, pp. 159-226 (1989). (GN 005396 – 5463)</p> <p><i>See generally</i> Gerald Salton, Michael McGill, <i>Introduction to Modern Information Retrieval</i>, McGraw-Hill, 1983, pp. 329-48.</p> <p><i>See generally</i> William B. Frakes, Ricardo Baeza-Yates, ed., <i>Information Retrieval / Data Structures & Algorithms</i>, Prentice Hall (1992) (GN 4430 – 60)</p>
6.	a first portion and a second portion	claims 1, 8, 13	<p>a first part separate and distinct from a second part</p> <p><u>Intrinsic Support:</u> ’593 patent at col. 2:3-2:12; col. 3:37-50; col. 4:60-5:29; col. 6:39-7:55; col. 10:25-51; col. 10:63-64; col. 11:24-44; col. 11:52-54, col. 12:6-33.</p> <p><u>Extrinsic Support:</u> “portion: a part of a whole, either separated from or integrated with it; segment.” <i>Random House Webster’s College Dictionary</i>, Random House, New York, 1991, p. 1052. (GN 300204)</p> <p>“The logical, plain meaning of 'first and second part' is that the item described must have two components: a first and a second. . . . The figure drawings . . .</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
	fragment transmitting a query message containing every said hashed query fragment to a respective one of said plurality of query nodes indicated by said first portion of said hashed query fragment	claim 13	U.S. Patent 4,811,199. (GN 7147 – 7162) <u>Extrinsic Support:</u> Kenneth Baclawski and J. Elliott Smith, <i>A Unified Approach to High-Performance, Vector-Based Information Retrieval</i> , March 21, 1994. (JAR 294 –313)
8.	using, by said query node, said second portion of said respective hashed query fragment to access data according to a local hash table located on said query node each said query node uses said second portion of said hashed query fragment to access data according to a local hash table located on said query node said query node, upon receipt of said query message, using said second portion of said hashed query fragment to access data according to a local	claim 1 claim 8 claim 13	each query node receiving a hashed query fragment uses the second portion of the hashed query fragment as a key value to identify the address of data according to a local hash table stored on that query node <u>Intrinsic Support:</u> '593 patent at Abstract; col. 2:13-18; col. 2:66-3:16; col. 3:17-4:7; col. 4:23-8:7; col. 8:45-9:60; col. 10:25-51; col. 11:24-44; col. 12:6-33; Fig. 1, Fig. 2. Figs. 3, 3a, 3b, 4, 5, 6, 7, 7a, 8, and 8a. '593 Prosecution History, May 14, 1997 Amendment at 2-3. U.S. Patent 4,811,199. (GN 7147 – 7162) <u>Extrinsic Support:</u> Kenneth Baclawski and J. Elliott Smith, <i>A Unified Approach to High-Performance, Vector-Based Information Retrieval</i> , March 21, 1994. (JAR 294 –313)

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
	hash table located on said query node		
9.	local hash table	claims 1, 8, 13	<p>a table resident on and unique to a particular query node in which the unique location of the information in the table is determined by hashing a key value</p> <p><u>Intrinsic Support:</u> '593 patent at col. 1:33-62; col. 3:26-4:7; col. 5:30-39; col. 6:1-27; col. 7:50-65; col. 10:25-51; col. 11:24-44; col. 12:6-33.</p> <p>'593 Prosecution History, May 14, 1997 Amendment at 5.</p> <p>Chaturvedi et al., <i>Scheduling the Allocation of Data Fragments in a Distributed Database Environment: A Machine Learning Approach</i>, IEEE TRANS ON ENG'G MGMT., vol. 41, no. 2, 1994, pp. 194-206. (GN 524 – 537)</p> <p>Houtsma et al., <i>Parallel Hierarchical Evaluation for Transitive Closure Queries</i>, IEEE Apr. 1991. (GN 4932 – 4941)</p> <p>U.S. Patent No. 4,811,199 (GN 7147 – 7162)</p> <p><u>Extrinsic Support:</u> “hashing: 1) a key-to-address transformation in which the keys determine the location of the data. (2) The process of applying a formula to a record key to yield a number that represents a disk address.” <i>Webster’s New World Dictionary of Computer Terms</i>, 4th ed., Prentice Hall, 1992, p. 187. (GN 300244)</p> <p>“hashing: A technique for arranging a set of items, in which a hash function is applied to the key of each item to determine its hash value. The hash value identifies each item’s primary position in a hash table, and if this position is already occupied, the item is inserted wither in an overflow table or in another</p>

Google, Inc.'s Proposed Construction & Evidentiary Support		
No.	Term	Claim(s)
		<p>available positioning the table.” Christopher Booth, ed. <i>The New IEEE Standard Dictionary of Electrical and Electronics Terms</i> (5th Ed.: Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 586. (GN 300215)</p> <p>“hashing: A technique for arranging a set of items, in which a hash function is applied to the key of each item to determine its hash value. The hash value identifies each item’s primary position in a hash table, and if this position is already occupied, the item is inserted wither in an overflow table or in another available positioning the table.” <i>IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries</i> (Inst. of Electrical & Electronics Engineers, Inc.), 1991, p. 99. (GN 300228)</p> <p>“hashing algorithm: An algorithm used to derive an address within a specified range from a key value. A hashing algorithm is used with a random file to determine the address of the block in which a given record should be stored.” Gunton, Tony, <i>A Dictionary of Information Technology and Computer Science</i> (2nd ed.) (Oxford, UK: NCC Blackwell Ltd.), 1993, p. 136. (GN 292980)</p> <p>“Section 6.3 treats digital searching, and Section 6.4 discusses an important class of methods called hashing techniques, based on arithmetic transformations of the actual keys.” Donald Knuth, <i>The Art of Computer Programming, Volume 3, Sorting and Searching</i>, Addison-Wesley, 1973, p. 390.</p> <p>“6.4: Hashing. So far we have considered search methods based on comparing the given argument K to the keys in the table, or using its digits to govern a branching process. A third possibility is to avoid all this rummaging around by doing some arithmetical calculation on K, computing a function f(K) which is the location of K and the associated data on the table. . . . These considerations lead to a popular class of search methods commonly known as hashing or scatter storage techniques. The verb “to hash” means to chop something up or to make a mess out of it; the idea in hashing is to chop off some aspects of the</p>

Google, Inc.’s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>key and to use this partial information as the basis for searching. We compute a hash function $h(K)$ and use this value as the address where the search begins.” Donald Knuth, <i>The Art of Computer Programming, Volume 3, Sorting and Searching</i>, Addison-Wesley, 1973, p. 506-07.</p> <p>“Prototype Specifications . . . The hash algorithm is taken from Knuth, volume 3, section 6.4.” (JAR 219469)</p> <p>“hash: An associative access technique that maps the key via a hash function uniformly among a partitioned set of hash buckets. A key search consists of hashing the key to a bucket address and then examining the small hash bucket for records with the desired key.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 120; <i>see also generally</i> pp. 831-851.</p> <p>“Associative access on a single relation can be supported by two types of access paths that use fundamentally different approaches to solving the problem of translating attribute values into tuple addresses. These approaches, which yield fundamentally different functionality, are generally referred to as hashing (key transformation) and key comparison.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 833.</p> <p>“Hashing is based on the idea of using the primary key value as a parameter to a function which returns the storage location of the tuple as a result. This is the very principle of hashing, and it has a number of interesting properties. If the transformation function can be kept simple—that is, if it does not need large data structures with search paths and routing data—then the access cost to retrieve a tuple via its key is minimal: take the key value, do some arithmetic, and find the tuple at the address delivered by the function. Ideal hash algorithms allow for one-access retrieval, which is to say that only the page</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>holding the tuple needs to be read when accessing via the primary key. The functional principle of simple hashing is to relate to key value of a tuple and the page number in which it is stored, through a predefined function; because of this, simple hashing is not just an access method, but also a file organization technique. (This was explained in Chapter 14.) As such, it needs page address spaces with special properties, as with become obvious during the detailed description. Hash-based access paths support only queries of the type (key-attribute = const). They do not efficiently support range predicates such as key-attribute between A and B. The attribute used for hashing typically is a primary key of the relation, but can also be used for nonunique attributes.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 833.</p> <p>“Key comparison comprises all methods for maintaining a dynamic search structure on the set of values in the key attribute. These values (or compressed versions of them) can be organized into tables, lists, trees, and so on, depending on the amount of data, the type of search operations to be supported, and the storage size that is available for maintaining the search structure. If such a technique is used for a primary access path, the entire tuple can be stored in search structure (see Section 14.4.5, on key-sequenced files, in Chapter 14); otherwise, it will contain pointers to the tuples (e.g., TIDs). Sorting a file along some attribute and keeping it sorted under updates is a very simple example of a search structure based on key comparison—in this case, it is a sequential sorted list of tuples. If records are clustered within blocks according to the key attribute values, searching and scanning in sorted order can be performed efficiently.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 833.</p> <p>“Algorithms to implement associative access come in two flavors. The first one, hashing, supports access based on value equality only; it uses a transformation function that turns the attribute value into a page address where</p>

Google, Inc.'s Proposed Construction & Evidentiary Support		
No.	Term	Claim(s)
		<p>the tuple can be found.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 835.</p> <p>“As in hash-based memory organizations, there has to be a function to transform the attribute value into a file address, where the tuple will (most likely) be found.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 835.</p> <p>“[A] good hash function H has to map primary key values, which are very unevenly distributed over a large value range, into a tuple address space that is much smaller (proportional to the number of existing tuples), such that the resulting addresses are evenly distributed over the address range.” Jim Gray, Andreas Reuter, <i>Transaction Processing: Concepts and Techniques</i>, Morgan Kaufman, 1993, p. 839.</p> <p>“[i]f the hash function $h(p)$ is properly chosen, then the hash values will be approximately uniformly distributed over their range.” (JAR 108483)</p> <p>Kenneth Baclawski and J. Elliott Smith, <i>A Unified Approach to High-Performance, Vector-Based Information Retrieval</i>, March 21, 1994 (JAR 294–313)</p> <p>Kenneth Baclawski and J. Elliott Smith, <i>KEYNET: Fast Indexing for Semantically Rich Information Retrieval</i>, December 7, 1993 (JAR 828–845)</p> <p>JAR0146229-146266</p> <p>“The examination node 102 then encodes each feature fragment of the object by using a predefined hashing function. Data in the system was previously stored locally on the various index nodes using this hashing function to generate an index to the data in the local database. Thus, the use of the same hashing</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>function to generate an index for data storage and to generate hashed feature fragments for an information object assures that (1) data is distributed uniformly over the index nodes of the routing search engine during the storing of data and (2) the feature fragments are scattered uniformly over the index nodes during the processing of an object.” U.S. Patent No. 6,192,364, col. 6:46-58. (GN 299812 – 831)</p> <p>“The home node 107 then encodes each feature of the query by using a predefined hashing function. Data in the system was previously stored locally on the various query nodes 109 using this hashing function to generate an index to the data in the local database. Thus, the use of the same hashing function to generate an index for data storage and to generate hashed probes for a data query assures that (1) data is distributed uniformly over the query nodes 109 of the search engine during the storing of data and (2) the probes are scattered uniformly over the query nodes 109 during the processing of a query.” U.S. Patent No. 6,424,973, col. 7:58-8:2. (GN299832 – 854)</p> <p>“The home node encodes each fragment of the query by using a predefined hashing function. The same hashing function preferably is also used in generating indexes to storage locations for storing data locally in local databases on the various query nodes. The use of the same hashing function to generate an index for data storage and to generate hashed probes for a query assures that data is distributed uniformly over the query nodes of the search engine during the storing of data, and that the probes are scattered uniformly over the query nodes during the processing of a query.” U.S. Patent No. 6,463,433, col. 11:18-29. (GN 299855 – 877)</p> <p>“Thus, the use of the same hashing function to generate an index for data storage and to generate hashed probes for an object assures that data is distributed uniformly over the index nodes 106 of the data warehouse during the storing of data.” U.S. Patent No. 6,470,333 Col. 7:36-41.</p>

Google, Inc.'s Proposed Construction & Evidentiary Support		
No.	Term	Claim(s)
		<p>“The home node 105 encodes each fragment of the query by using a predefined hashing function. Data in the distributed computer database system was previously stored locally on the various index nodes 112 using this hashing function to generate an index to the data in the local database. In particular, if a fragment includes a link then it is hashed and stored as a link fragment, while if a fragment does not include a link then it is hashed and stored as an index fragment. Thus, the use of the same hashing function to generate an index for data storage and to generate hashed probes for a query assures that 1. data is distributed uniformly over the index nodes of the search engine during the storing of data and 2. the probes are scattered uniformly over the index nodes during the processing of a query.” U.S. Patent No. 6,505,191, col. 8:61-9:8. (GN 299917 – 940)</p> <p>“The examination node 102 then encodes each feature fragment of the object by using a predefined hashing function. Data in the system was previously stored locally on the various index nodes using this hashing function to generate an index to the data in the local database. Thus, the use of the same hashing function to generate an index for data storage and to generate hashed feature fragments for an information object assures that (1) data is distributed uniformly over the index nodes of the routing search engine during the storing of data and (2) the feature fragments are scattered uniformly over the index nodes during the processing of an object.” U.S. Patent No. 6,535,881, col. 6:59-7:3. (GN 299898 – 916)</p> <p>U.S. Patent No. 6,505,191 Prosecution History, July 3, 2002 Response to Office Action at 5. (GN 299941 – 300172)</p> <p><i>See generally</i> Gerald Salton, <i>Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer</i>, Addison-Wesley, pp. 159-226 (1989). (GN 005396 – 5463)</p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p>See generally Gerald Salton, Michael McGill, <i>Introduction to Modern Information Retrieval</i>, McGraw-Hill, 1983, pp. 329-48.</p> <p>See generally William B. Frakes, Ricardo Baeza-Yates, ed., <i>Information Retrieval / Data Structures & Algorithms</i>, Prentice Hall (1992). (GN 4430 – 60)</p>
10.	<p>returning, by each said query node</p> <p>each said query node ... returns</p> <p>said query node ... returning</p>	<p>claim 1</p> <p>claim 8</p> <p>claim 13</p>	<p>each query node that accesses data returns an object identifier to the home node</p> <p><u>Intrinsic Support:</u> '593 patent at Abstract; col. 1:11-13; col. 3:51-4:7; col. 4:22-36; col. 7:50-65; col. 10:25-51; col. 11:24-44; col. 12:6-33; Fig. 1, Fig. 2.</p> <p>'593 Prosecution History, Dec. 11, 1996 Amendment at p. 8-9; April 16, 1996 Office Action at 3; June 7 1996 Response to Office Action at 11, 14.</p> <p><u>Extrinsic Support:</u> Kenneth Baclawski and J. Elliott Smith, <i>A Unified Approach to High-Performance, Vector-Based Information Retrieval</i>, March 21, 1994 (JAR 294–313)</p>
11.	predetermined degree of relevance	claims 3, 9	<p>a predefined degree of similarity; only results meeting or exceeding a predetermined level are returned to the user after the object identifier has been returned</p> <p><u>Intrinsic Support:</u> '593 patent at col. 2:13-2:18; col. 3:51-4:21; col. 8:16-20; col. 8:45-53; col. 10:55-60; col. 11:45-48; Fig. 2, Fig. 8, Fig. 8a.</p> <p>'593 Prosecution History, June 7, 1996 Amendment at p. 16-17.</p> <p>Chaturvedi et al., <i>Scheduling the Allocation of Data Fragments in a Distributed</i></p>

Google, Inc.'s Proposed Construction & Evidentiary Support			
No.	Term	Claim(s)	
			<p><i>Database Environment: A Machine Learning Approach</i>, IEEE TRANS ON ENG'G MGMT., vol. 41, no. 2, 1994, pp. 194-206. (GN 524 – 537)</p> <p><u>Extrinsic Support:</u></p>