Name of the Subject: Distributed Systems	
Assign. No	**Name of the Assignment along with the description like language to be used if applicable, sample input, expected output, observations or assumptions, references etc.**
1.	**Optimizing Join query in Distributed Database** This project proposes an algorithm which attempts to predict the best execution plan for a join query retrieving data from two remote computers. The goal of this algorithm is to execute efficiently the join query in a distributed database. The algorithm estimates the response time for the possible execution plan based on the size of the transmission data, the transmission speed, and the machine processing speed. The Unified Process model can be used for the development activities. The project conducts a test on data transmission. The test result shows that the total response time of data transmission in my system is comprised of the time of querying, transferring, and inserting. The times of querying, transferring and inserting are linear and have a strong correlation with the amount of transmission data. The parallel query process can save the time of executing the SQL query which retrieves data from remote sites. A test of join query performance indicates that the size of transmission data and the transmission speed have impacts on the total response time. By comparing the estimated data size and estimated response time with the actual execution results, we can discover that the proposed algorithm can predict the best plan for most of the join queries with an exception of join queries selecting only a small amount of data.
2.	**ODBC Connections with the Data source** 1. Create an Application 2. Allocating the Environment Handle 3. Declaring the Application's ODBC Version 4. Choosing a Data Source or Driver 5. Allocating a Connection Handle 6. Connection Attributes 7. Establishing a Connection 8. Driver Manager Connection Pooling

	9. Run an Application 10. Disconnecting from a Data Source or Driver 11. The Driver Manager's Role in the Connection Process
3.	**Borg Collective.** In Star Trek, the Borg Collective is a kind of mindshare - tell one Borg something and they all know it. Logically, the collection of all agents in an agent system might be viewed as a Borg Collective if there is a way for any agent to query all the agents to know what they know. But there is another puzzle - a human should not receive all this information else they will be overwhelmed by the blizzard of information so they should just be told what they need to know. The general puzzle then is to provide agents with limited knowledge of exactly what they need to know but give them a query capability that somehow can access all information. One solution approach is a central database of all information that all agents know. A drawback of one central database is a single point of failure so if this central database agent dies then agents have no one to ask for additional information. Also, if say Group 1 (Rangers agents) stored their group knowledge in one central repository and Group 2 (local police agents) used another, then can one group query across the other group's organizational boundaries if the need arises - when both groups are part of one mission. Another solution is they broadcast peer to peer. Or maybe hubs. The problem is to analyze a collection of such architectures and determine pros and cons of different possibilities. Problem not fully defined. Potential MS or PhD. • A subproblem might be, if Agent1 wants to send messages TO: <any agent passing some predicate, e.g., any agent within 500 feet of me -or- the nearest medic who is available -or- all rangers who report to squad leader Bill -or- any person whose heartbeat is irregular and who has high blood pressure>, then how can this kind of location or logical query be evaluated on the fly using the agent network>. Where and how to these queries get evaluated. What if some of the query can be evaluated locally and some via other agents?
4.	**Views of a Simulation** Let's assume that every message that any agent sends to another agent is logged (by bcc) to a central DBMS. Then that log can be viewed as a representation of one or a collection of scenarios or vignettes. Furthermore, it can be replayed in its entirety to watch the entire action unfold again (with some caveats - agents in simulation mode do not send messages to each other, instead all messages originate in the log and all messages sent by agents in simulation mode are logged/ignored/deleted (as well as any actions an agent might take during this dream-like replay). A first puzzle is to implement simulation model (mostly done). A second puzzle is to

	implement a subsetting capability so views or subsets of simulations can be replayed, e.g., just those messages in which a medic and a wounded soldier are replayed. [This might involve also selecting other messages received from other agents during this time in case those side effect the agent-to-agent conversations one is studying.] The ability to subset simulations seems to be new!]
5.	**Mission Editor** Let's assume a mission is specified using an ontology editor (e.g., Stanford Protege) consisting of a set of EiA agents, messages they can send, and other mission specific tasks. • One problem is, can this be done in faster than wall clock time so the system can be used to create a reasonable collection of agents for an evolving mission. Slower than wall clock time means you are too late for any real mission. A constraint is, one should think of this as always editing because, even during missions, it should be possible to download new initialization information at any time, including new capabilities (e.g., .jar file agent plugins). • Another problem is, can you design the mission in a centralized environment and then push a button and have mission init messages sent to provision a collection of generic agents that are now specialized to work on this mission. How can you maintain consistency between the central ontology and the distributed agents?
5.	**Resynchronizing an Agent who has been disconnected** If an agent becomes disconnected so that at some later time it receives a queued set of messages, how does it update itself, take a subset of the actions it should take, but not act on intermediate messages that are not longer up to date (e.g., a message received an hour ago that an enemy is hiding in a bush - but now friendly forces have already taken the bush and are hiding there). A similar problem is how to add an agent into the mix and bring them up to date on the current situation.
6.	**Agent Ontology Interoperability** Assume you develop a ranger ontology using Protege and then a Police ontology. Each understand somewhat different missions and messages. But then you are asked to form a coalition where both can interoperate. What is involved in making this interoperability easier to manage.
7.	**Genomics Algebra: A New Integrating Data Model, Language, and Tool for Processing and Querying Genomic Information** The dramatic increase of mostly semi-structured genomic data, their heterogeneity and high variety, and the increasing complexity of biological applications and methods mean that many and very important challenges in biology are now challenges in computing and here especially in databases. In contrast to the many query-driven approaches advocated in the literature, we propose a new integrating approach that is based on two fundamental pillars. The *Genomics Algebra* provides an extensible set of high-level *genomic data*

	types (*GDT*) (e.g., *genome*, *gene*, *chromosome*, *protein*, *nucleotide*) together with a comprehensive collection of appropriate *genomic functions* (e.g., *translate*, *transcribe*, *decode*). The *Unifying Database* allows us to manage the semi-structured contents of publicly available genomic repositories and to transfer these data into GDT values. These values then serve as arguments of Genomics Algebra operations which are supposed to be embedded into a DBMS query language.
8.	**Design and Implementation of an Image Algebra and its Integration into Database Systems** Multimedia database systems are of interest in many application areas which deal with video, image, audio, text, or graphic data, or any kind of mixture of them. The goal of this topic is to focus exclusively on the image part. We then call these systems *image database systems*. Images are of particular interest in many applications since they allow the visual transport of large volumes of information in a packed manner. Although a large knowledge about images exists from a processing standpoint in disciplines like computer graphics, computer vision, and image processing, a study of the literature reveals that not much is known about the conceptual view the user has or should have on image database systems. Simply collecting images in a database and enabling to browse them does not justify the use of a database system. Some central questions are: What kind of interface should these systems provide to the user? What kind of query languages should be made available? What are the central operations on images? How can images be represented so that they can support the identified operations in an efficient way? Are formats like jpeg, tiff, and many others appropriate for this purpose? Our idea is to incorporate the answers to all these questions into a type system (we call it *algebra*) for images. That is, the first task is to design and implement AN IMage ALgebra called ANIMAL, which provides data types and operations for images. The next step is then to integrate these types and operations into an extensible database system (like Oracle, DB 2, or others) and its query language, and thus to create an image database system. Later, extensions are conceivable with respect to image indexing and information retrieval.
9.	**Spatial and Spatio-Temporal Data Warehousing** Research in data warehousing and on-line analytical processing (OLAP) has produced important technologies for the design, management, and use of information systems for decision support. However, despite the continued success and maturing of the field, much work remains to be done in the future. Given the wealth of models, terminology, and definitions, the first task is to review the most important models and their treatment of the basic concepts including the notions “dimension”, “fact”, “hierarchy”, “data cube”, and many more. The intent should be to evaluate existing models based on their expressiveness, flexibility, separation of modeling aspects from implementation aspects, etc. With the knowledge gained, the second task is to develop an overall and comprehensive

	conceptual model adapted to the users' needs and abstracting from implementation aspects. The third task is to identify existing OLAP operators, to get an overview of their capabilities, and to learn how they can be used to manipulate multi-dimensional data (e.g., cube, roll-up, drill-across). The fourth task is to define these OLAP operators and perhaps new ones, which have so far not been considered, on the basis of the designed model in task 2. The fifth task is to identify new applications for data warehousing and OLAP in the spatial and spatio-temporal domain and to extend the model of task 2 correspondingly. The impact of new, advanced, and non-standard data types on the data warehousing concepts has to be explored. Also, new OLAP operations have to be detected and formally defined. The sixth task is to implement the complete model as a *data warehouse extension package* that can be integrated as a cartridge, datablade, or extender into Oracle, DB2, and Informix.
10.	**Design and Implementation of Spatial Graphs (Networks) and their Integration into Database Systems** An important spatial concept in maps are spatial graphs representing, for example, road networks, railway networks, and power networks. This PhD project has three goals. First, a design of an *abstract model*should give a definition of spatial graphs and their properties and further identify the most important operations and predicates on them. An example of an important operation on spatial graphs is to find the *shortest path*from a source to a destination. Such a model will be based on mathematical concepts like point set theory, point set topology, graph theory, and functions. Second, since the abstract model is based on infinite point sets and functions and cannot be directly implemented, a *discrete model* is needed that yields finite representations for the infinite concepts of the abstract model and algorithms for the operations and predicates of the abstract model. Third, the ultimate goal is to incorporate spatial graphs into databases systems and their query languages.
11.	**A Video Database Management System for Advancing Video Database Research**
12.	**PIC Based Intelligent Tracking System Using Solar Power**
13.	**GPS-GSM Integration for Enhancing Public Transportation Management Services**
14.	**Handling concurrency in Distributed Databases**
15.	**Employment Exchange**

16.	**Deadlock Detection Algorithms**
17.	**Web based Image Sharing Portal** In this project you are going to develop a web-based database application system that provides service to its clients for storing, sharing, and searching their photos. The system, similar to flickr and PhotoShelter, can be used by its clients to * upload and store photos; * enter and update the descriptive information (time, place, persons, caption, series, owner, copyright) for photos; * specify the access privileges for your friends and/or public to share your photos; and * search for photos with given words and/or other specified conditions.
18.	**CourseRank** CourseRank is a social tool that helps students make informed choices about courses and take advantage of the available learning options. It displays official university information and statistics, such as website course descriptions, grade distributions, and results of official course evaluations. Students can anonymously rank courses they have taken, add comments, and rank the accuracy of each others' comments. They can also get recommendations, organize their classes into a semester schedule or devise a four year plan and track their progress. CourseRank also functions as a feedback tool for faculty and administrators, ensuring that information is as accurate as possible.
19.	**Mashups** You can build interesting web based DBMS applications by aggregating feeds from multiple web sites. Here are some examples: http://pipes.yahoo.com/pipes/ http://code.google.com/gme/
20.	**Integrated Inventory Management System** A Department has its own inventory managed manually by the store. The Store handles purchase of all items (consumable and non-consumable) and issue to respective laboratories (software, hardware, etc) and individual (faculty, staff and students). Laboratories can also issue items to individuals. Department runs various projects, each project will have separate inventory management system which is not related to the departmental inventory. The projects are managed by the faculties in the department and any faculty can have more than one projects with separate inventory management system for each project. Inventory Management System consists of the following processes:

	Purchase: Purchase process starts with requisition from any faculty, staff and students (Students and staff should forward their requisition by their respective Prof-in-charges, otherwise the requisition is not valid). Storekeeper would go through all the requisitions and forward only the valid ones to the HOD. HOD would allow the purchase of required items according to availability of funds. In projects PI (Principle Investigator) is the HOD and requisitions are sent directly to the PI. HOD decides the mode of purchase (i.e. either by quotation or directly). Direct purchase has a limitation of amount (i.e. if a single item costs more than Rs. 10,000 then one have to go through quotation and purchase order). Quotations are invited from various vendors (at least 6) and vendor opting for the lowest bid is selected and hence purchase order is issued against the corresponding vendor. The vendor supplies the item/(s) which is recorded in the books and bill is submitted along the installation certificate (if required). Bills are verified finally by HOD. Book Transfer: Sometimes the Institute purchases items and issued to the department. Department in this case behaves like a lab. Issue: The procured items could be issued to the person who has sent the requisition or respective labs who can further issue the same to individuals. Once it is non-consumable then it can be issued and reissued several times or can be maintained in a lab. Once any item (PC or a Server) is used for a particular special reason (mail server, webserver, etc) is to be recorded. Search: Searching facilities for a particular item (by Make or Company, Specification, Short Description, Year of Purchase, Year/Month of Warranty/AMC expiry, Type of Items, etc.). Repairs and Upgrades: All the upgrades and repairing done to a particular system are recorded in the database. This is updated by the respective labs and storekeeper does not have the privilege for the same. Damaged and Write-off: The items which are not in use for a long time due to the fact that the system cannot be repaired and have to be written-off (Stock clearance).
21.	**Web based Tour Planner** Design a tour planner agent that offers the end users with a list of best tour plans against user provided budget and tour options. The tour options may include the places of visit, the mode of journey, hotel booking etc
22.	**Personal Information Management System** The PIM system manages the activities people perform in order to acquire, organize, maintain, retrieve and use information items such as documents (paper-based and digital), web pages and email messages for everyday use to complete tasks (work-related or not) and

	fulfill a person's various roles (as parent, employee, friend, member of community, etc.). One ideal of PIM is that we always have the right information in the right place, in the right form, and of sufficient completeness and quality to meet our current need. Technologies and tools such as personal information managers help us spend less time with time-consuming and error-prone activities of PIM (such as looking for information).
23.	**Census Database** The 2011 Census of India happened on February 9, 2011. The goal of this project is to build a database which can provide various informations based on the census data. The parameters recorded in census can be found in this (and other) site. http://censusindia.gov.in/Metadata/Metada.htm The system may be interfaced with geographical maps like Google API.
24.	**Data Stream Processing for Network Data** Build a sophisticated network monitoring tool using streaming data algorithms. The Lawrence Berkeley Laboratory has some wide-area TCP traces available here. You are also free to collect your own data by instrumenting computer equipment to obtain network traffic traces, video game control messages (great for spatial data), etc. Make sure to respect the privacy of any persons involved.
25.	**Massive Multiplayer Online Games** MMO is one of the common workloads of modern processors and clouds. Refresh rate of such games are often in the order of 30Hz. These maybe posed as important database problems. Seethe foll. reference for interesting directions. http://www.cs.cornell.edu/johannes/papers/2007/2007-SIGMODRecord-Games.pdf We can attempt the subproblem of game state recovery. Massively multiplayer online games (MMOs) are persistent virtual worlds that allow tens of thousands of users to interact in fictional settings. Users typically select a virtual avatar and collaborate with other users to solve puzzles or complete quests. These games are extremely popular, and successful MMOs have millions of subscribers and have generated billions of dollars in revenue. Unlike single player computer games, MMOs must persist across user sessions. Players can leave the game at any time, and they expect their achievements to be reflected in the world when they rejoin. Similarly, it is unacceptable for the game to lose player data in the event of a crash. These demands make it essential for MMOs to ensure that their state is durable. As such, MMO developers have been forced to develop ad-hoc solutions or invest in expensive special purpose hardware to achieve some degree of fault tolerance. The goal of this project is to adapt database recovery algorithms for this purpose. Here are some references and a simulator of the workload. http://www.cs.cornell.edu/bigreddata/games/recovery.php

26.	**Graph Databases** Many modern applications like social networking sites use graph data. The goal of this project is to build a web based interface with a backend database for benefit of users in handling graph data. Here is an example graph database: http://neo4j.org/ And source of some graph data and algos which may be interfaced with the DB: http://snap.stanford.edu/
27.	**Spatial Databases** Build spatial database applications using geometric queries. Here is an open source spatial sql engine. http://postgis.refractions.net/ Example projects are (i) Location based services (e.g., restaurants near you), (ii) Evacuation planning.
28.	**Distributed Database** Study on open source distributed database, e.g., the Voldemort System used by LinkedIn. http://project-voldemort.com/ Interface the system with other distributed filed systems more suitable for inverted index search like Hadoop.
29.	**XML Databases** Write a simple parser and query processor for restricted Xquery language which can return results on a small xml database. You can use the DBLP XML dataset as a benchmark.
30.	**Object oriented Database** Use an OODBMS to implement any of the Information System Design projects described above. You may use any open source OODBMS like http://www.ozone-db.org/frames/home/what.html
31.	**External Sorting Algorithms** Implement external sorting algorithms and study their performance on your hardware configuration.
32.	**Distributed 2PL** write a program to simulate the two phase locking protocol on a distributed database.
33.	**Higher dimensional index data structures for multimedia databases** The goal of this project is to implement higher dimensional data structures like R-Tree and KD-Trees and use them for multimedia (e.g., music, video) databases.
34.	**MARIPOSA: A WIDE-AREA DISTRIBUTED DAT ABASE SYSTEM**

35.	**Design a Database Storage Mechanisms** e.g. Cassandra, Google BigTable
36.	**Mini Search Engine** This project designs and implements a mini search engine. You are probably familiar with Google, Altavista or Yahoo, which are some of the most popular search engines that you can find on the Web. The task performed by a search engine is, as the name says, to search through a collection of documents. Given a set of texts and a query, the search engine will locate all documents that contain the keywords in the query. The problem may be therefore reduced to a search problem, which can be efficiently solved with the use of data structures. The project is to design and implement an algorithm that searches a collection of documents. There will be a set of 50 documents and a set of sample queries. First, the documents will be processed and their content (i.e. words / tokens) will be stored in the data structures (in information retrieval, this phase is called ***indexing***). Next, for every input query, the query will be processed and its keywords will be searched in the documents, using the previously implemented data structures and an algorithm (this phase is called ***retrieval***). For each such query, the documents that satisfy the query are displayed. The queries may contain simple Boolean operators, that is AND and OR, which act in a similar manner with the well known analogous logical operators. For instance, a query: "*Keyword1 AND Keyword2*" should retrieve all documents that contain both these keywords (elements). "*Keyword 1 OR Keyword 2*" instead will retrieve documents that contain either one of the two keywords. **Example** Consider the following sample documents. *Doc1:* I like the class on data structures and algorithms. *Doc2:* I hate the class on data structures and algorithms. *Doc3:* Interesting statistical data may result from this survey. Here are the answers to some queries: *Query 1:* data Doc1, Doc2, Doc3 *Query2:* data AND structures Doc1, Doc2 *Query 3:* like OR survey

	Doc1, Doc3 HOW IT WOULD WORK Format of the documents is looked at. Input is parsed. The punctuation is already separated from the words, so there is no need to worry about that. One word at a time is read and added to the data structure. As data structures, following may be considered : dictionaries / hashtables, trees – such as 2-4 trees, AVL trees, balanced binary search trees. For every word, a list of documents where it occurs is stored, in order to allow for efficient searches and Boolean operators later on.

Format for the long assignment: Please provide the assignment in the following template.

Signature of the Faculty