Shri Vile Parle Kelavani Mandal's

DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)

Academic Year (2022-23)

Year: 3 Semester: V

Program: T.Y. B. Tech. (Computer Engineering)

Subject: Advanced Database Management System

Date:

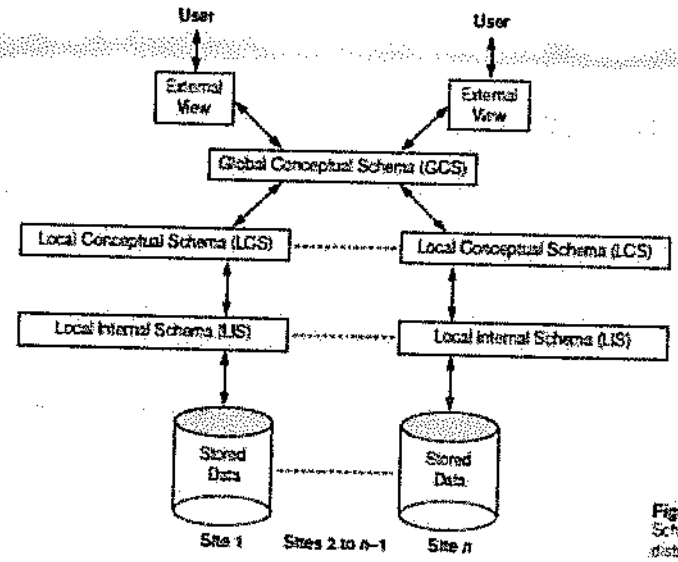
Max. Marks: 75

Time: 10: 30 am to 1:30 pm

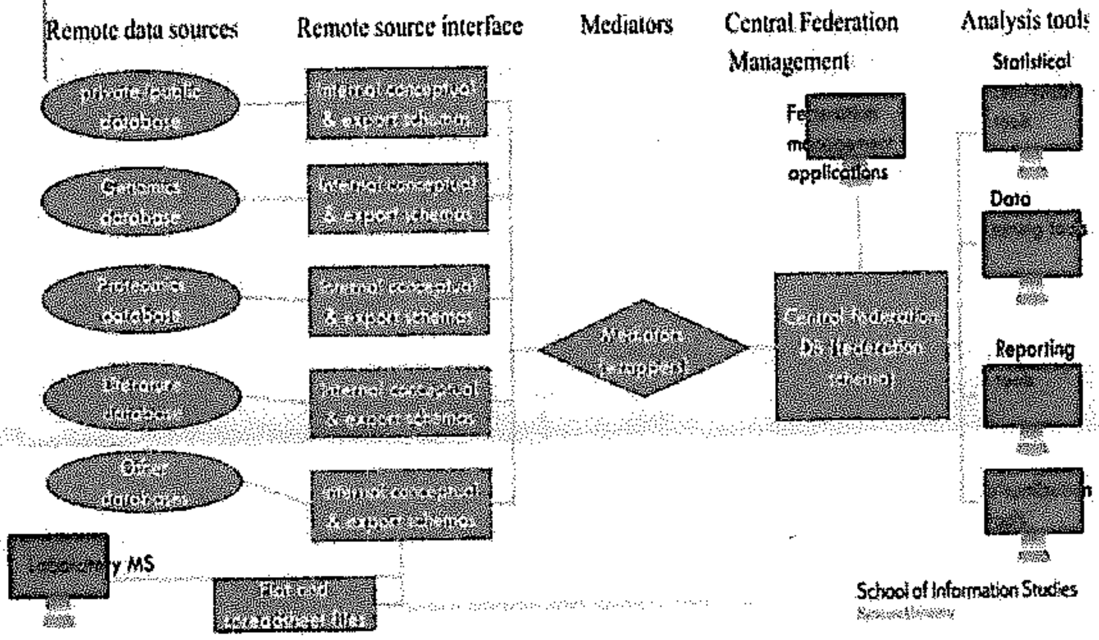
Duration: 3 Hours

REGULAR EXAMINATION

ANSWER KEY

Question No.		Max. Marks
Q1 (a)	 **Figure 23.6** Schema architecture of distributed databases. A distributed database system allows applications to access data from local and remote databases. In a homogenous distributed database system, each database is an Oracle Database. In a heterogeneous distributed database system, at least one of the databases is not an Oracle Database. Distributed databases use a client/server architecture to process information requests. This section contains the following topics: 1. Homogenous Distributed Database Systems 2. Heterogeneous Distributed Database Systems OR	[10]

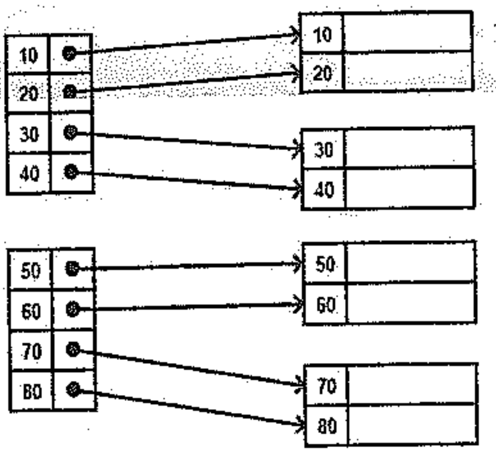
Federated database architecture



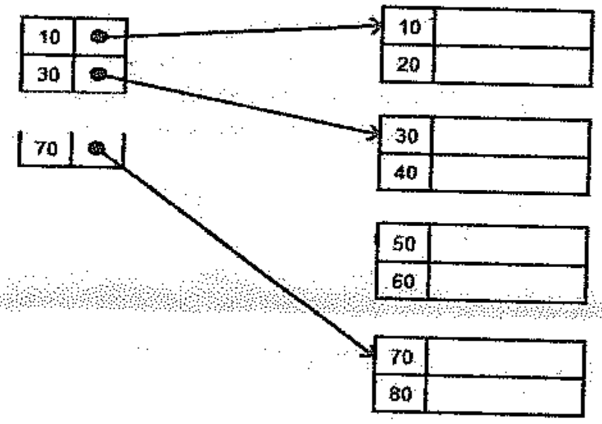
- A **federated database system** is a type of meta-database management system (DBMS), which transparently integrates multiple autonomous database systems into a single **federated database.**
- The constituent databases are interconnected via a computer network and may be geographically decentralized. Since the constituent database systems remain autonomous, a federated database system is a contrastable alternative to the (sometimes daunting) task of merging together several disparate databases.
- A federated database, or **virtual database**, is the fully integrated, logical composite of all constituent databases in a federated database system.

Q1 (b) [05]

- Dense index — Index record appears for every search-key value in the file.



- Sparse Index: contains index records for only some search-key values.
  - Applicable when records are sequentially ordered on search-key
- To locate a record with search-key value *K* we:
  - Find index record with largest search-key value < *K*
  - Search file sequentially starting at the record to which the index record points.



OR

- Many factors contribute to time cost

disk accesses, CPU, or even network *communication*

- Number of seeks
- Number of blocks read
- Number of blocks written

Q2 (a) [10]

- Algorithm **A1 (linear search)**. Scan each file block and test all records to see whether they satisfy the selection condition.
  - Cost estimate = b_r block transfers + 1 seek= $t_s + b_r * t_T$
    b_r denotes number of blocks containing records from relation *r*
  - Linear search can be applied regardless of
    - selection condition or
    - ordering of records in the file, or
    - availability of indices
- Algorithm **A2 (binary search)- Applicable if file is ordered** on an attribute and the selection condition is an equality condition on the attribute
  - Cost estimate = $\lceil \log 2(b_r) \rceil * (t_T + t_s)$
- **A3 (primary index, equality on key)**. Retrieve a single record that satisfies the corresponding equality condition
  - $Cost = h_i * (t_T + t_S) + (t_T + t_S) = (h_i + 1) * (t_T + t_S)$
- **A4 (primary index, equality on nonkey)** Retrieve multiple records.
  - Records will be on consecutive blocks, Let b = number of blocks containing matching records $Cost = h_i * (t_T + t_S) + t_S + t_T * b$
- **A5 (secondary index, equality on nonkey)**.

- Retrieve a single record if the search-key is a candidate key *Cost* = $(h_i + 1) * (t_T + t_S)$
- Retrieve multiple records if search-key is not a candidate key each of n matching records may be on a different block

  Cost = $h_i * (t_T + t_S) + n * (t_T + t_S) = (h_i + n) * (t_T + t_S)$

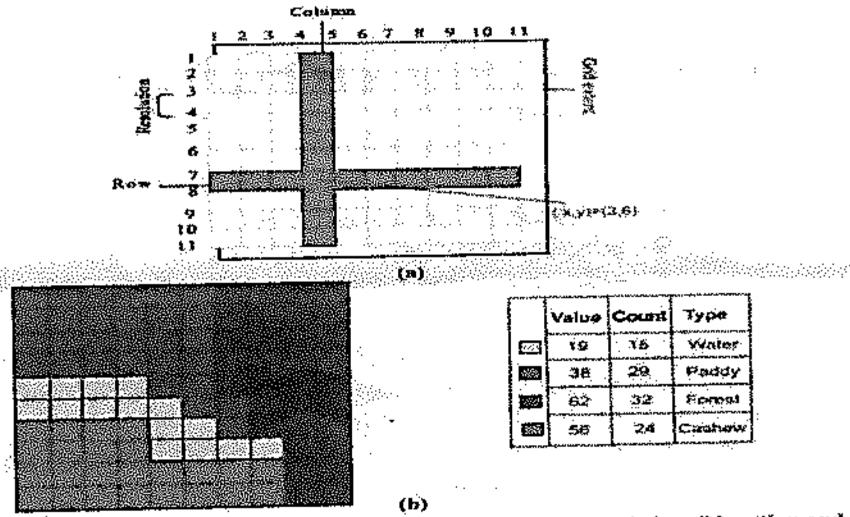
OR

- There are two different aspects of time in temporal databases.
- **Valid Time:** Time period during which a fact is true in real world, provided to the system.
- **Transaction Time:** Time period during which a fact is stored in the database, based on transaction serialization order and is the timestamp generated automatically by the system.
- Temporal Relation is one where each tuple has associated time; either valid time or transaction time or both associated with it.
- **Uni-Temporal Relations:** Has one axis of time, either Valid Time or Transaction Time.
- **Bi-Temporal Relations:** Has both axis of time – Valid time and Transaction time. It includes Valid Start Time, Valid End Time, Transaction Start Time, Transaction End Time.

Q2 (b) [05]

Raster Data Models:

- The raster data model is composed of a regular grid of cells in specific sequence and each cell within a grid holds data. The conventional sequence is row by row which may start from the top left corner.
- In this model, basic building block is the cell. The representation of the geographic feature in this model is used by coordinate, and every location corresponds to a cell. Each cell contains a single value and is independently addressed with the value of an attribute.
- One set of cells and associated value is a layer. Cells are arranged in layers. A data set can be composed of many layers covering the same geographical areas e.g., water, paddy, forest, cashew.



	Value	Count	Type
	19	16	Water
	38	29	Paddy
	62	32	Forest
	56	24	Cashew

Fig. 5.3: Illustration of raster data; (a) raster grid matrix with their cell location and coordinates, and (b) raster grid and its attribute table

OR

JSON	XML
JSON object has a type	XML data is typeless

	JSON types: string, number, array, Boolean	All XML data should be string	
	Data is readily accessible as JSON objects	XML data needs to be parsed.	
	JSON is supported by most browsers.	Cross-browser XML parsing can be tricky	
	JSON has no display capabilities.	XML offers the capability to display data because it is a markup language.	
	JSON supports only text and number data type.	XML support various data types such as number, text, images, charts, graphs, etc. It also provides options for transferring the structure or format of the data with actual data.	
	Retrieving value is easy	Retrieving value is difficult	
	Supported by many Ajax toolkit	Not fully supported by Ajax toolkit	
	A fully automated way of deserializing/serializing JavaScript.	Developers have to write JavaScript code to serialize/de-serialize from XML	
	Native support for object.	The object has to be express by conventions – mostly missed use of attributes and elements.	
	It supports only UTF-8 encoding.	It supports various encoding.	
	It doesn't support comments.	It supports comments.	
	JSON files are easy to read as compared to XML.	XML documents are relatively more difficult to read and interpret.	
	It does not provide any support for namespaces.	It supports namespaces.	
	It is less secured.	It is more secure than JSON.	
Q3 (a)	Π ename, salary(σ dname="Computer"(Emp ⋈ Dept)) Π ename, salary (Emp ⋈ σ (dname=" Computer"(Dept))) Π ename, salary (Emp ⋈ Πdid (σ (dname=" Computer"(Dept)))) Draw tree for each		[10]

Q3 (b) [05]

S. No.	Relational Model	Document Model
1.	It is row-based.	It is document-based.
2.	Not suitable for hierarchical data storage.	Generally used for hierarchical data storage.
3.	It consists of a predefined schema.	It consists a dynamic schema.
4.	ACID properties are followed by this model. (Atomicity, Consistency, Isolation, and Durability).	CAP theorem are followed by this mc Partition tolerance).
5.	It is slower .	It is faster than Relational Model.
6.	Supports complex joins.	Does Not support for complex joins.
7.	It is column-based.	It is field-based.
8.	They are vertically scalable	They are horizontally scalable
9.	Fast replication support is not provided.	They provide easy replication support
10.	It is more used now-a-days to store data in database.	It is comparatively less used.

11.	To add attributes in a relational model, database schema needs to be modified for including additional columns and their data types.	In a document-based database, you need to add additional key-value pairs in the document for representing new fields.	

Q4 (a)	a) With simple join Case 1; Send employee relation from site 1 to site 3 and send department relation from site 2 to site 3 and then compute result at site 3 74,500 bytes Case 2; Send employee relation from site 1 to site 2, compute result at site 2 and send result from site 2 to site 3. 85,000 bytes Case 3; Send Department relation from site 2 to site 1, compute result at site 1 and send result from site 1 to site 3. 45,500 bytes. b) With semi-join Case 1; Send employee relation with attributes ename,salary and did from site 1 to site 3 and send department relation with attributes did and dname from site 2 to site 3 compute result at site 3 38,000 bytes Case 2; Send employee relation with attributes ename,salary and did from site 1 to site 2, compute result at site 2 and send result from site 2 to site 3. 56,000 bytes Case 3; Send Department relation with attributes did and dname from site 2 to site 1, compute result at site 1 and send result from site 1 to site 3. 38,000 bytes. OR Fragmentation is a process of dividing the whole or full database into various sub tables or sub relations so that data can be stored in different systems. The small pieces of sub relations or sub tables are called fragments. • Horizontal fragmentation • Vertical fragmentation • Mixed or Hybrid fragmentation	[10]
Q4 (b)	1. The objects may be complex, or they may consists of low-level object □ However, to represent the data of these complex objects through relational database models you would require many tables – at least one each for each inherited class and a table for the base class. □ In order to ensure that these tables operate correctly we would need to set up referential integrity constraints as well. □ On the other hand, object oriented models would represent such a system Object Oriented Database very naturally through, an inheritance hierarchy. Thus, it is a very natural choice for such complex objects.	[05]

	2. Consider a situation where you want to design a class, (let us say a Date class), the advantage of object oriented database management for such situations would be that they allow representation of not only the structure but also the operation on newer user defined database type such as finding the difference of two dates. ☐ Thus, object oriented database technologies are ideal for implementing such systems that support complex inherited objects, user defined data types 3. Another major reason for the need of object oriented database system would be the seamless integration of this database technology with object-oriented applications. Software design is now, mostly based on object oriented technologies. Thus, object oriented database may provide a seamless interface for combining the two technologies. 4. The Object oriented databases are also required to manage complex, highly interrelated information. They provide solution in the most natural and easy way that is closer to our understanding of the system. Michael Brodie related the object oriented system to human conceptualisation of a problem domain which enhances communication among the system designers, domain experts and the system end users. 5. An object oriented database is presently being used for various applications in areas such as, e-commerce, engineering product data management; and special purpose databases in areas such as, securities and medicine. 6. Some of the major concerns of object oriented database technologies include access optimization, integrity enforcement, archive, backup and recovery operations etc.	
Q5 (a)	The typical method of enforcing discretionary access control in a database system is based on the granting and revoking privileges 1. Privilege granting- Grant < privilege list> on EMP_table to User1; GRANT ALL PRIVILEGES ON Emp_Salary TO PUBLIC; Grant update(amount) on loan to U1, U2,U3; grant update on branch to U1 with grant option; 2. Privilege revocation- revoke select on branch from U1,U2,U3 revoke grant option for select on branch from U1	[10]
Q5 (b)	**Raster Data Models:** • The raster data model is composed of a regular grid of cells in specific sequence and each cell within a grid holds data. The conventional sequence is row by row which may start from the top left corner. • In this model, basic building block is the cell. The representation of the geographic feature in this model is used by coordinate, and every location corresponds to a cell. Each cell contains a single value and is independently addressed with the value of an attribute. One set of cells and associated value is a layer. Cells are arranged in layers. A data set can be composed of many layers covering the same geographical areas e.g., water, paddy, forest, cashew. **Vector Data Models:** • Vector data model comprises discrete features. Features can be discrete locations or events (points), lines, or areas (polygons). This model uses the geometric objects of point, line and polygon. • In vector model, the point is the fundamental object. Point represents anything that can be described as a discrete x,y location (e.g., hospital, temple, well, etc.). Line or polyline (sequence of lines) is created by connecting the sequence of points. • End points are usually called as nodes and the intermediate points are termed as vertices	[05]