CMR INSTITUTE OF TECHNOLOGY	USN	

Internal Assesment Test – I, September 2019

Sub:	DATABASE MANAGEMENT SYSTEM							Code:	18MCA31
Date:	06-09-2019	Duration:	90 mins	Max Marks:	50	Sem:	III	Branch:	MCA

Note: Answer any full 5 questions. All questions carry equal marks. Total Marks: 50

Question	Marks	OBE CO	OBE RBT
PART-I			
1. Explain with necessary diagram the Database system environment architecture.	10	CO1	L1
(OR)			
2. List and explain the Advantages of DBMS approach.	10	CO1	L1
PART-II			
3. Explain about the Centralized and client/server architecture with diagram.	10	CO1	L1
(OR)			
4. Describe about the following: (i) actors on the scene (ii) database languages and interfaces	10	CO1	L1
PART-III			
5. Define the following terms: (i) Join (ii) Division (iii) Cartesian Product (iv) Union (v) Set Difference	10	CO1	L1
(OR)			
6. What are integrity constraints? Discuss the various update operations on relations and the type of integrity constraints that must be checked for each update operation.	10	CO1	L2
PART-IV			
7 Design an ER diagram for the following scenario. The ABC COMPANY PVT LTD., database keeps track of a company's employees, departments, and projects. The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have several locations. A department controls a number of projects, each of which has a unique name, a unique number, and a single location. We store each employee's name, Social Security number, address, salary, sex (gender), and birth date. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. We keep track of the current number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each employee (who is another employee). We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent's first name, sex, birth date, and relationship to the employee.	10	CO1	L6
(OR)			
8 Define the term relational algebra expression. Explain in detail about complete set of relational algebra operations with examples.	10	CO1	L1
PART-V			
9 Illustrate unary relational operations, with appropriate syntax and example.	10	CO1	L1
(OR)			
10 Demonstrate ER-to-mapping algorithm with an example.	10	CO1	L1

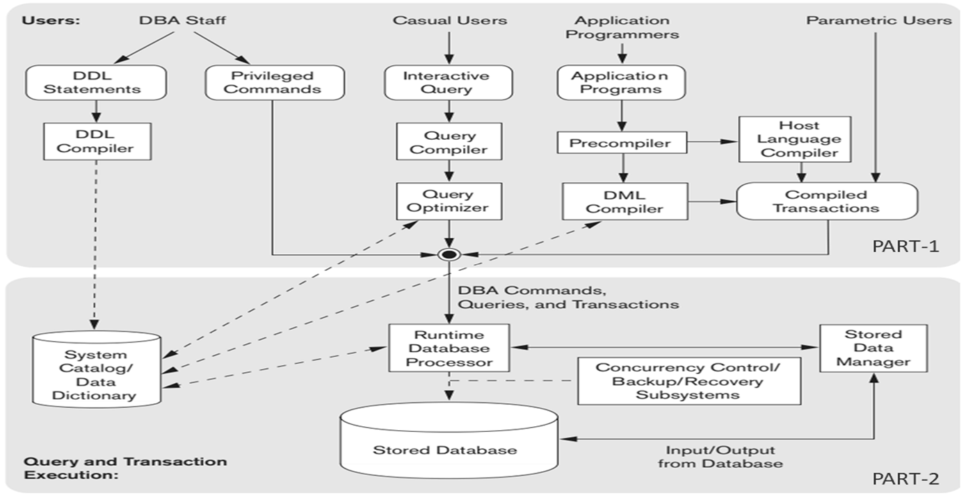
CMR Institute of Technology, Bengalore – 560 037
Department of Computer Applications
Scheme and Solutions for Internal Assessment Test – I

18MCA31 – DATABASE MANAGEMENT SYSTEM **Semester / Section: III** **Date of Test: 06-09-2019**

---

1. Explain with necessary diagram the Database system (10) environment architecture.

- A DBMS is a complex software system
- discuss the types of software components that constitute a DBMS and the types of computer system software with which the DBMS interacts
- two parts:
  - The top part: various users and their interfaces
  - The lower part: storage of data and processing of transactions
- Access to the disk is controlled primarily by the operating system (OS)
- buffer management module to schedule disk read/write, because this has a considerable effect on performance
- A higher-level stored data manager module: controls access to DBMS information that is stored on disk, whether it is part of the database or the catalog
- DDL Statement: used by the DBA for defining the database and tuning it
- DDL Compiler: processes schema definitions and stores descriptions of the schemas in the DBMS catalog
- Privileged commands: The commands which are exclusively used by the DBA
- Interactive query: The interface by which the casual users and persons with occasional need for information from the database interact
- Query compiler: Queries are parsed and validated for correctness of the query syntax
- Query optimizer: Responsible for optimising the query and its execution



Part-1: users and their interfaces Part-2: Internals of the DBMS

- Pre-compiler: Extracts DML commands from an application program written in a host programming language
- DML compiler: Compilation of pre-compiled DMLC commands into object code for database access
- Host language compiler: The rest of the program is sent to the host language compiler.
- Compiled transactions: Canned transactions are executed repeatedly by parametric users, who simply supply the parameters to the transactions

Scheme: Diagram: 4 marks and Explanation: 6 Marks

---

2. List and explain the Advantages of DBMS approach. (10)

i. Controlling Redundancy

- redundancy in storing the same data multiple times leads to several problems
  - there is the need to perform a single logical update
  - storage space is wasted when the same data is stored repeatedly
  - Files that represent the same data may become inconsistent

ii. Restricting Unauthorized Access

- A DBMS should provide a security and authorization subsystem
- most users will not be authorized to access all information in the database

iii. Providing Persistent Storage for Program Objects

- Databases can be used to provide persistent storage for program objects and data structures

iv. Providing Storage Structures and Search Techniques for Efficient Query Processing

- the DBMS must provide specialized data structures and search techniques to speed up disk search for the desired records – index
- The DBMS often has a buffering or caching module that maintains parts of the database in main memory buffers
- The query processing and optimization module of the DBMS is responsible for choosing an efficient query execution plan for each query based on the existing storage structures.

v. Providing Backup and Recovery

- responsible for recovery
- the database is restored to the state it was in before the transaction started executing

vi. Providing Multiple User Interfaces

- a DBMS should provide a variety of user interfaces
  - query languages for casual users,
  - programming language interfaces for application programmers,
  - forms and command codes for parametric users, and
  - menu-driven interfaces and natural language interfaces for standalone users

vii. Representing Complex Relationships among Data

- A DBMS must have the capability to represent a variety of complex relationships among the data, to define new relationships as they arise, and to retrieve and update related data easily and efficiently

viii. Enforcing Integrity Constraints

- The simplest type of integrity constraint involves specifying a data type for each data item
- referential integrity constraint: specifying that a record in one file must be related to records in other files
- Primary key constraint: uniqueness on data item values

ix. Permitting inferencing and Actions Using Rules: A trigger is a form of a rule activated by updates to the table, which results in performing some additional operations to some other tables, sending messages, and so on.

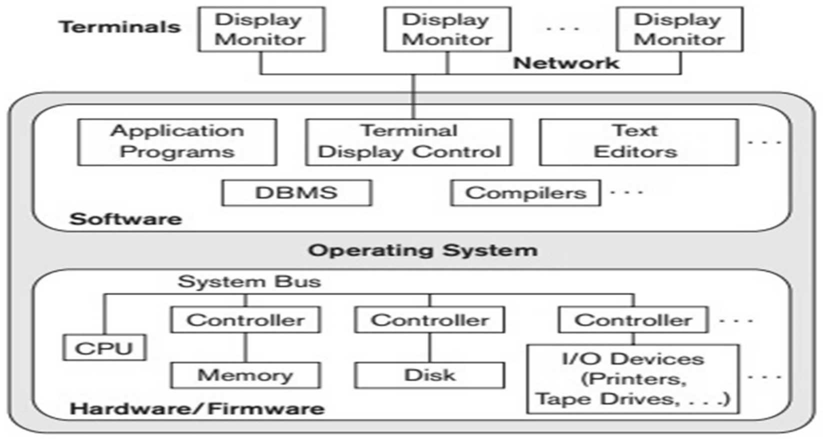
x. Additional Implications of Using the Database Approach

- Potential for Enforcing Standards: The DBA can enforce standards in a centralized database environment
- Reduced Application Development Time: Development time using a DBMS is estimated to be one-sixth to one-fourth of that for a traditional file system
- Flexibility: Modern DBMSs allow certain types of evolutionary changes to the structure of the database without affecting the stored data and the existing application programs
- Availability of Up-to-Date Information: As soon as one user's update is applied to the database, all other users can immediately see this update
- Economies of Scale: reduce the wasteful overlap activities thereby reducing the overall costs of operation and management

Scheme: One advantage carries one mark (10 X 1 =10 Marks)

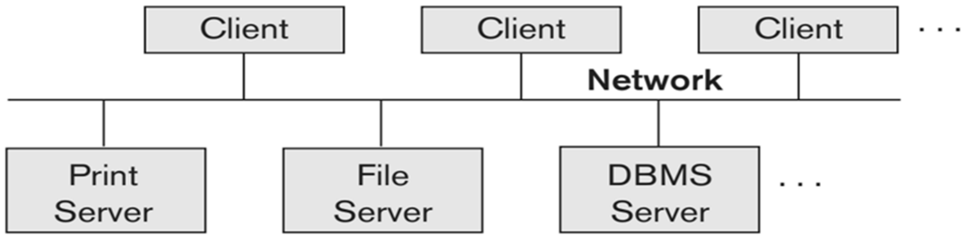
3. Explain about the Centralized and client/server architecture with diagram. (10)

- Earlier architectures used mainframe computers to provide the main processing for all system functions, including user application programs and user interface programs, as well as all the DBMS functionality
- centralized DBMS in which all the DBMS functionality, application program execution, and user interface processing were carried out on one machine
- all processing was performed remotely on the computer system
- only display information and controls were sent from the computer to the display terminals

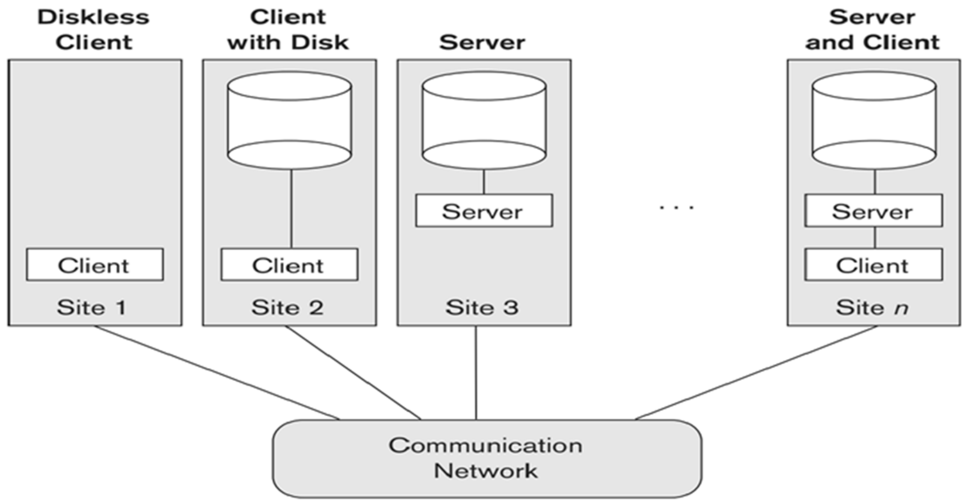


- Basic Client/Server Architectures
- define specialized servers with specific functionalities
- A client is typically a user machine that provides user interface capabilities and local processing
- A server is a system containing both hardware and software that can provide services to the client machines, such as file access, printing, archiving, or database access
- The client/server architecture was developed to deal with computing environments in which a large number of PCs, workstations, file

servers, printers, data base servers, Web servers, e-mail servers, and other software and equipment are connected via a network



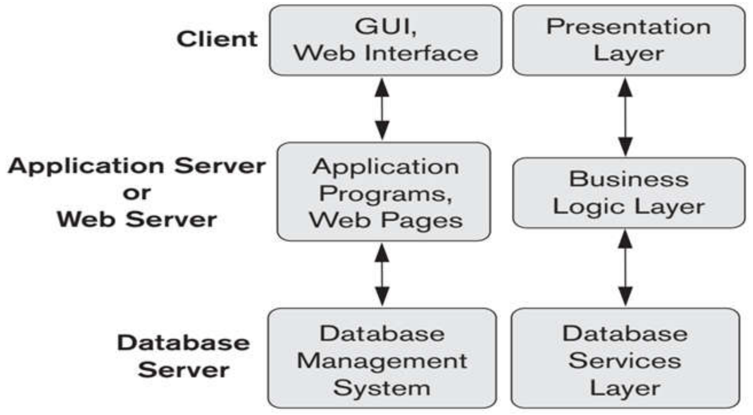
Logical two-tier client/server architecture



Physical two-tier client/server architecture

- Two main types of basic DBMS architectures were created on this underlying client/server framework: two-tier and three-tier
- Two-Tier Client/Server Architectures for DBMSs:
- the software components are distributed over two systems: client and server
  - client side: has the user interface and application programs
  - server side: the query and transaction functionality related to SQL processing
- The different approach to two-tier client/server architecture was taken by some object-oriented DBMSs
- the client level: handle the user interface; data dictionary functions; DBMS interactions with programming language compilers; global query optimization, concurrency control, and recovery across multiple servers; structuring of complex objects from the data in the buffers; and other such functions
- the server level: the DBMS software responsible for handling data storage on disk pages, local concurrency control and recovery, buffering and caching of disk pages, and other such functions
- Advantages: simplicity and seamless compatibility with existing systems
- Three-Tier and n-Tier Architectures for Web Applications
- The emergence of the Web changed the roles of clients and servers, leading to the three-tier architecture

- adds an intermediate layer (middle tier – application layer or web server) between the client and the database server
- runs the application programs and storing business rules (procedures or constraints) that are used to access data from the database server
- The presentation layer: displays information to the user and allows data entry.
- The business logic layer: handles intermediate rules and constraints before data is passed up to the user or down to the DBMS.
- The middle layer can also act as a Webserver, which retrieves query results from the database server and formats them into dynamic Web pages that are viewed by the Web browser at the client side
- The bottom layer includes all data management services.



Scheme: Diagrams: 4 marks
Types of client/server architectures: 6 Marks

4. Describe about the following: **(10)**
(i) actors on the scene
(ii) database languages and interfaces.

(i) Actors on the Scene:

- the people whose jobs involve the day-to-day use of a large database

i. Database Administrators: responsible for authorizing access to the database, coordinating and monitoring its use, and acquiring software and hardware resources as needed

ii. Database Designers: responsible for identifying the data to be stored in the database and for choosing appropriate structures to represent and store this data.

- develop views of the database that meet the data and processing requirements of these groups.

iii. End Users: the people whose jobs require access to the database for querying, updating, and generating reports

- several categories of end users:
  - Casual end users: occasionally access the database, but they may need different information each time (middle or higher-level managers)
  - Naive or parametric end users: constantly querying and updating the database (bank tellers, reservation agents)
  - Sophisticated end users: include engineers, scientists, business analysts, and others who thoroughly familiarize themselves with the facilities of the DBMS in order to implement their own applications to meet their complex requirements.

- Standalone users: maintain personal databases by using ready-made program packages that provide easy-to-use menu-based or graphics-based interfaces.

iv. System Analysts and Application Programmers

- System analysts determine the requirements of end users, especially naive and parametric end users, and develop specifications for standard canned transactions that meet these requirements.
- Application programmers implement these specifications as programs; then they test, debug, document, and maintain these canned transactions.

(ii) Database languages and interfaces

- The DBMS must provide appropriate languages and interfaces for each category of users
- DBMS Languages

i. Data Definition Language (DDL): used by the DBA and by database designers to define conceptual and internal schemas
ii. Storage Definition Language (SDL): used to specify the internal schema
iii. View Definition Language (VDL): specify user views and their mappings to the conceptual schema

- The DBMS provides a set of operations or a language called the Data Manipulation Language (DML) to manipulate the database
- There are two main types of DMLs:
- A high level non-procedural language to specify complex database operations (Example: SQL)

(Eg. SQL)

- follows row-at-a-time processing

ii. A low-level or procedural DML must be embedded in a general-purpose programming language. (Eg. PL/SQL)

- DBMS Interfaces: User-friendly interfaces provided by a DBMS may include the following:
- *Menu-Based Interfaces for Web Clients or Browsing (Pull down menus)*
  - Menus do away with the need to memorize the specific commands and syntax of a query language
  - *Forms-Based Interfaces:* displays a form to each user, can fill out all of the form entries to insert new data, or they can fill out only certain entries
  - *Graphical User Interfaces:* displays a schema to the user in diagrammatic form
  - utilize both menus and forms
  - *Natural Language Interfaces:* accept requests written in English or some other language and attempt to understand them
  - *Speech Input and Output:* Limited use of speech as an input query and speech as an answer to a question or result of a request (Eg. Telephone directory information)
  - The speech input is detected using a library of predefined words and used to set up the parameters that are supplied to the queries.

Scheme: Actors on the scene: 5 Marks and Interfaces: 5 Marks

5. Define the following terms: (i) Join (ii) Division (iii) Cartesian Product (iv) Union (v) Set Difference. **(10)**

(i) Join: Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition

(ii) Division: Produces a relation R(X) that includes all tuples t[X] in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z=X\cup Y$

(iii) Cartesian Product: Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2

(iv) Union: Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible

(v) Set Difference: Produces a relation that includes all the tuples in R_1 that are not in R_2; R_1 and R_2 must be union compatible

Scheme: each definition carries 2 marks (5 X 2 = 10 Marks)

6. What are integrity constraints? Discuss the various update operations on relations and the type of integrity constraints that must be checked for each update operation. **(10)**

- A relational database schema S is a set of relation schemas S = {R_1, R_2, ..., R_m} and a set of integrity constraints (IC)
- The referential integrity constraint is specified between two relations and is used to maintain the consistency among tuples in the two relations
- the referential integrity constraint states that a tuple in one relation that refers to another relation must refer to an existing tuple in that relation
- There are three basic operations that can change the states of relations in the database: Insert, Delete, and Update (or Modify)
  - Insert is used to insert one or more new tuples in a relation,
  - Delete is used to delete tuples, and
  - Update (or Modify) is used to change the values of some attributes in existing tuples
- The Insert Operation
  - Insert can violate any of the Four types of constraints / violations:
    i. Domain constraints can be violated if an attribute value is given that does not appear in the corresponding domain or is not of the appropriate data type
    ii. Key constraints can be violated if a key value in the new tuple t already exists in another tuple in the relation r(R)
    iii. Entity integrity can be violated if any part of the primary key of the new tuple t is NULL
    iv. Referential integrity can be violated if the value of any foreign key in t refers to a tuple that does not exist in the referenced relation
- The Delete Operation
- can violate only referential integrity
- occurs if the tuple being deleted is referenced by foreign keys from other tuples in the database

- To specify deletion, a condition on the attributes of the relation selects the tuple (or tuples) to be deleted
- Several options are available if a deletion operation causes a violation.

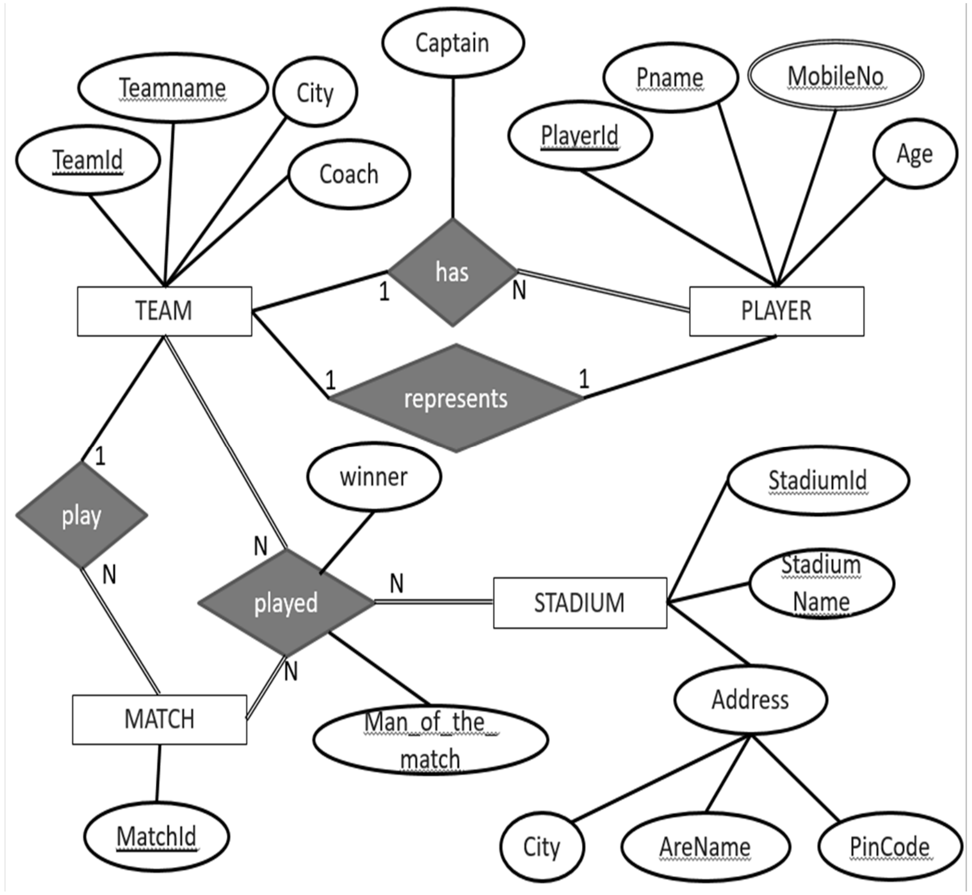
(i) restrict: reject the deletion.
(ii) Cascade: to attempt to cascade (or propagate) the deletion by deleting tuples that reference the tuple that is being deleted

(iii) set null or set default: modify the referencing attribute values that cause the violation each such value is either set to NULL (except primary key attribute) or changed to reference another default valid tuple

- The update operation:
- used to change the values of one or more attributes in a tuple (or tuples) of some relation R
- It is necessary to specify a condition on the attributes of the relation to select the tuple (or tuples) to be modified

Scheme:
Definition of integrity constraints: 2 Marks
Explanation of Operations and violations: 8 Marks

7. Design an ER diagram for the following scenario. (10)
The ABC COMPANY PVT LTD., database keeps track of a company's employees, departments, and projects. The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have several locations. A department controls a number of projects, each of which has a unique name, a unique number, and a single location. We store each employee's name, Social Security number, address, salary, sex (gender), and birth date. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. We keep track of the current number of hours per week that an employee works on each project. We also keep track of the direct supervisor of each employee (who is another employee). We want to keep track of the dependents of each employee for insurance purposes. We keep each dependent's first name, sex, birth date, and relationship to the employee.



Scheme:

Diagram: 6 Marks Step by Step explanation: 4 Marks

8. **Define the term relational algebra expression. Explain in detail about complete set of relational algebra operations with examples** **(10)**

- A sequence of relational algebra operations forms a relational algebra expression, whose result will also be a relation that represents the result of a database query

I. Unary relational Operations: Select and Project

i. The SELECT Operation

- used to choose a subset of the tuples from a relation that satisfies a selection condition
- the SELECT operation is denoted by: $\sigma_{<selection\ condition>}(R)$
  - Eg.: $\sigma_{salary>30000}$ (EMPLOYEE)

ii. The PROJECT operation (π)

- selects certain columns from the table and discards the other columns
- the result of the PROJECT operation can be visualized as a vertical partition of the relation into two relations:
  - one has the needed columns (attributes) and contains the result of the operation, and
  - the other contains the discarded columns
- The general form of the PROJECT operation is: $\pi_{<attribute\ list>}(R)$
  - Eg.: $\pi_{Lname,\ Fname,\ Salary}$ (EMPLOYEE)

II. Relational algebra operations from set theory:

- Two relations R(A1, A2, ..., An) and S(B1, B2, ..., Bn) are said to be union compatible (or type compatible) if they have the same degree n and if dom(Ai) = dom(Bi) for 1 ≤ i ≤ n.
- This means that the two relations have the same number of attributes and each corresponding pair of attributes has the same domain.
- UNION: The result of this operation, denoted by R ∪ S, is a relation that includes all tuples that are either in R or in S or in both R and S. Duplicate tuples are eliminated.
- INTERSECTION: The result of this operation, denoted by R ∩ S, is a relation that includes all tuples that are in both R and S.
- SET DIFFERENCE (or MINUS): The result of this operation, denoted by R – S, is a relation that includes all tuples that are in R but not in S
- UNION and INTERSECTION are:
  - commutative operations: that is, R∪S = S∪R and R∩S = S∩R
  - associative operations: that is, R∪(S∪T) = (R∪S)∪T and (R∩S)∩T = R∩(S∩T)
  - The MINUS operation is not commutative: R − S ≠ S − R

Example:

STUDENT							
Fn	Susan	Ramesh	Johnny	Barbara	Amy	Jimmy	Ernest
Ln	Yao	Shah	Kohler	Jones	Ford	Wang	Gilbert

INSTRUCTOR					
Fname	John	Richardo	Susan	Francis	Ramesh
Lname	Smith	Browne	Yao	Johnson	Shah

STUDENT ∪ INSTRUCTOR

Fn	Ln
Susan	Yao
Ramesh	Shah
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert
John	Smith
Ricardo	Browne
Francis	Johnson

STUDENT ∩ INSTRUCTOR

Fn	Ln
Susan	Yao
Ramesh	Shah

INSTRUCTOR - STUDENT

Fname	Lname
John	Smith
Ricardo	Browne
Francis	Johnson

STUDENT - INSTRUCTOR

Fn	Ln
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

- Cartesian Product (also known as CROSS PRODUCT or CROSS JOIN, denoted denoted by X)
- the relations on which it is applied do not have to be union compatible
- this set operation produces a new element by combining every member (tuple) from one relation (set) with every member (tuple) from the other relation (set)
- the result of $R(A_1, A_2, ..., A_n) \times S(B_1, B_2, ..., B_m)$ is a relation Q with degree n+m attributes $Q(A_1, A_2, ..., A_n, B_1, B_2, ..., B_m)$, in that order
- The resulting relation Q has one tuple for each combination of tuples
  - if R has n_R tuples (denoted as $|R| = n_R$), and S has n_S tuples, then R×S will have $n_R * n_S$ tuples

III. Binary relational operations: Join and Division

- The JOIN Operation

- used to combine related tuples from two relations into single "longer" tuples
- The JOIN operation can be specified as a CARTESIAN PRODUCT operation followed by a SELECT operation
- The general form of a JOIN operation on two relations R(A_1, A_2, ..., A_n) and S(B_1,B_2, ..., B_m) is R $\bowtie_{\text{<join condition>}}$ S
- ▪ Equi Join
- join conditions with equality comparisons
- the only comparison operator used is =
- equality join condition specified on the two attributes requires the values to be identical in every tuple in the result.
- ▪ Natural Join
- requires that the two join attributes (or each pair of join attributes) have the same name in both relations
- Allows the combination of relations that are associated by a foreign key
- The DIVISION Operation (denoted by ÷, applied to two relations)
- R(Z) ÷ S(X) where X subset Z. Let Y = Z – X, let Y be the set of attributes of R that are not attributes of S
- The result of division is a relation T(Y) that includes a tuple t if tuples t_R appear in R with t_R [Y] =t, and with t_R[X] =t_S for every tuple t_S in S
- For a tuple t to appear in the result T of the DIVISION operation, the values in t must appear in R in combination with every tuple in S

Scheme: Each category of operations: 5 Marks

9. Illustrate unary relational operations, with appropriate syntax and example. (10)

(i) The SELECT Operation (σ)

- used to choose a subset of the tuples from a relation that satisfies a selection condition
- the SELECT operation is denoted by: $\sigma_{\text{<selection condition>}}$ (R)
  - Eg.: $\sigma_{\text{salary>30000}}$ (EMPLOYEE)
- The Boolean expression specified in <selection condition> is made up of a number of clauses of the form:
  - <attribute name> <comparison op> <constant value> or
  - <attribute name> <comparison op> <attribute name>
- (Eg.) $\sigma_{\text{(Dno=4 AND Salary>25000) OR (Dno=5 AND Salary>30000)}}$(EMPLOYEE)
- The Boolean conditions AND, OR, and NOT have their normal interpretation, as follows:
  - (cond1 AND cond2) is TRUE if both (cond1) and (cond2) are TRUE; otherwise, it is FALSE.
  - (cond1 OR cond2) is TRUE if either (cond1) or (cond2) or both are TRUE; otherwise, it is FALSE.
  - (NOT cond) is TRUE if cond is FALSE; otherwise, it is FALSE.
- a sequence of SELECTs can be applied in any order.
- In addition, we can always combine a cascade (or sequence) of SELECT operations into a single SELECT operation with a conjunctive (AND) condition;
- that is,

$\sigma_{<cond1>}(\sigma_{<cond2>}(...(\sigma_{<condn>}(R)) ...))=\sigma_{<cond1> \text{ AND} <cond2> \text{ AND...AND } <condn>}(R)$

(ii) The PROJECT operation (π)

- selects certain columns from the table and discards the other columns
- the result of the PROJECT operation can be visualized as a vertical partition of the relation into two relations:
  - one has the needed columns (attributes) and contains the result of the operation, and
  - the other contains the discarded columns
- The general form of the PROJECT operation is: $\pi_{<attribute\ list>}(R)$
  - Eg.: $\pi_{\text{Lname, Fname, Salary}}$ (EMPLOYEE)
- The PROJECT operation removes any duplicate tuples, so the result of the PROJECT operation is a set of distinct tuples (duplicate elimination)
- $\pi_{\text{Sex, Salary}}$(EMPLOYEE) is equivalent to

select DISTINCT sex, salary from EMPLOYEE;

Scheme: Select operation: 5 Marks Project operation: 5 marks

10. Demonstrate ER-to-mapping algorithm with an example. (10)

Step-1: Mapping of Regular Entity Types

- For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E.
- Choose one of the key attributes of E as the primary key for R.
- If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.

Step-2: Mapping of Weak Entity Types

- For each weak entity type W (Child Entity) in the ER schema with owner entity type E (Parent Entity), create a relation R and include all simple attributes (or simple components of composite attributes) of W as attributes of R.
- In addition, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s).
- The primary key of R is the combination of the primary key(s) of the owner(s) (Strong or Parent Entity) and the partial key of the weak entity (Child Entity) type W, if any.

Step-3: Mapping of Binary 1:1 Relation Types

- identify the relations S and T that correspond to the entity types participating in R
  - There are three possible approaches:

(1) Foreign Key approach: Choose the relations with total participation in R – say S - and include T's primary key in S as a foreign key.

(2) Merged relation option: An alternate mapping of a 1:1 relationship type is possible by merging the two entity types and the relationship into a single relation. This may be appropriate when both participations are total.

(3) Cross-reference or relationship relation option: The third alternative is to set up a third relation W(T.primarykey, S.primaryKey)

- for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types.

Step-4: Mapping of Binary 1:N Relationship Types

- identify the relation S that represent the participating entity type at the N-side of the relationship type.
- Include a foreign key in S the primary key of the relation T that represents the other entity type participating in R.
- Include any simple attributes of the 1:N relation type as attributes of S.

Step-5: Mapping of Binary M:N Relationship Types

- create a new relation S to represent R.
- Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; their combination will form the primary key of S.
- Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S

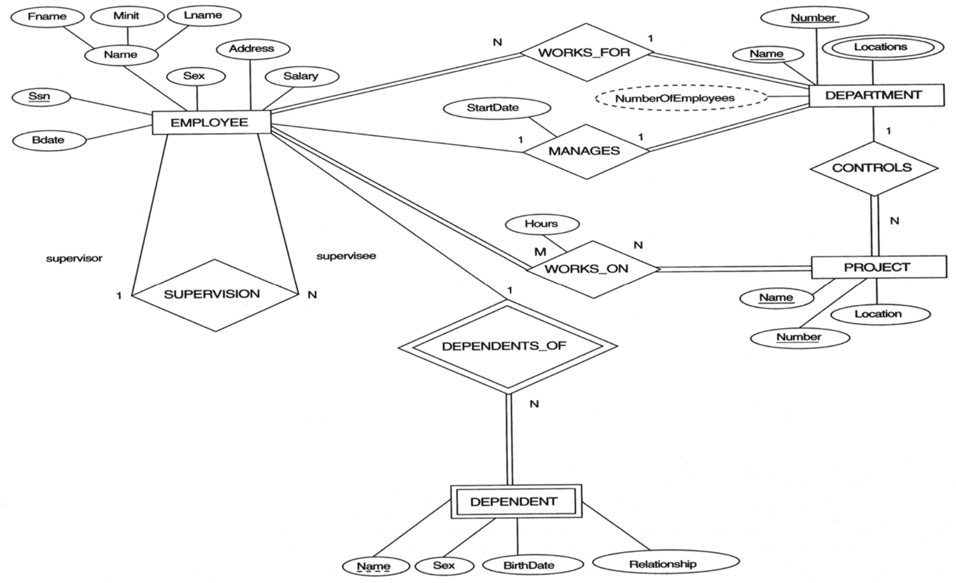
Step-6: Mapping of Multivalued attributes

- For each multivalued attribute A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type that has A as an attribute.
- The primary key of R is the combination of A and K.
- If the multivalued attribute is composite, we include its simple components.

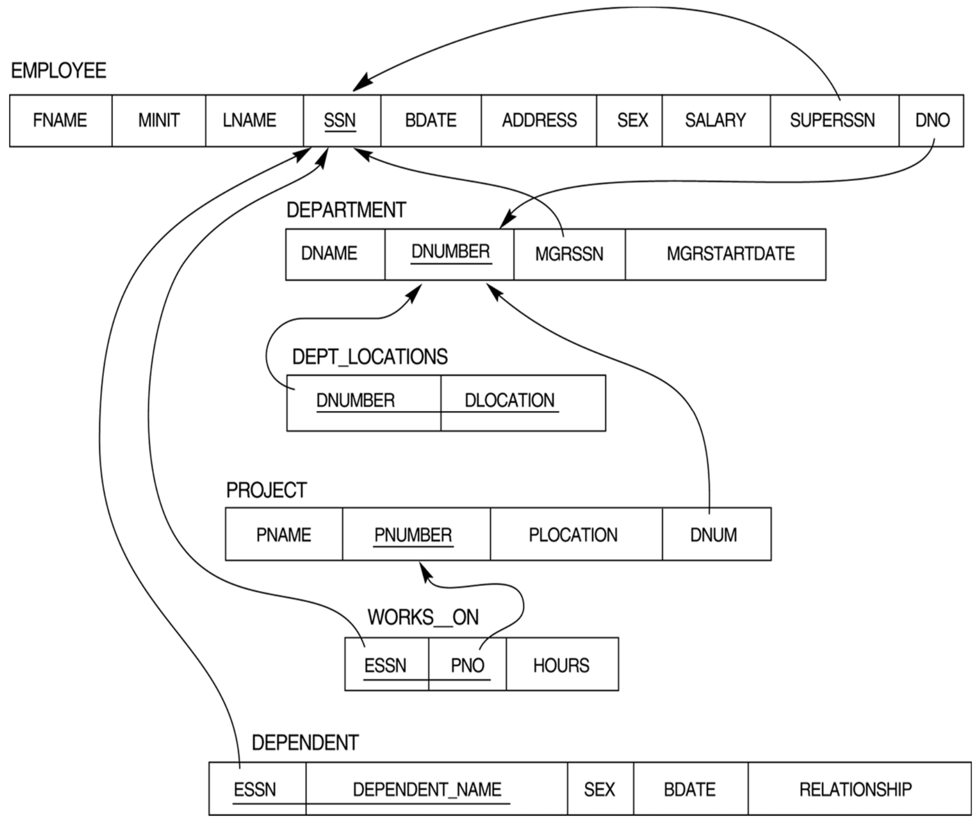
Step-7: Mapping of N-ary Relationship Types

  - For each n-ary relationship type R, where n>2, create a new relationship S to represent R.
  - Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.
- Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S.

Example: The ER conceptual schema diagram



Result of mapping the ER schema into a relational schema


EMPLOYEE
FNAME
MINIT
LNAME
SSN
BDATE
ADDRESS
SEX
SALARY
SUPERSSN
DNO
DEPARTMENT
DNAME
DNUMBER
MGRSSN
MGRSTARTDATE
DEPT_LOCATIONS
DNUMBER
DLOCATION
PROJECT
PNAME
PNUMBER
PLOCATION
DNUM
WORKS__ON
ESSN
PNO
HOURS
DEPENDENT
ESSN
DEPENDENT_NAME
SEX
BDATE
RELATIONSHIP