USN | | | | | | | | | | |

Internal Assessment Test 1 – Nov 2021

Sub:	Database Management System					Sub Code:	18CS53	Branch:	CSE
Date:	12/11/21	Duration:	90 mins	Max Marks:	50	Sem / Sec:	5/A,B,C	OBE	

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1	Explain the component modules of DBMS and their interaction, with the help of a diagram.	[10]	CO1	L2
2	Write an ER diagram for hospital management considering at least four entities.	[10]	CO1	L3
3	Explain the syntax of a SELECT,INSERT,DELETE,UPDATE,ALTER statement in SQL With example	[10]	CO3	L2
4	Discuss the following (Concept, example and ER notation): i) Weak entity & Identifying relationship ii) Participation Constraints iii) Recursive Relationships and Role names. iv) Cardinality Ratio	[10]	CO1	L2
5 (a)	List operation of relational algebra and explain the purpose of each with examples.	[06]	CO2	L2
(b)	Differentiate between Strong entity set and weak entity set.	[04]	CO1	L2
6 (a)	Define an entity and an attribute. Explain the different types of attributes that occur in an ER model, with an example.	[06]	CO1	L2
(b)	Discuss the main characteristics of database approach. How it differ from traditional file system.	[04]	CO1	L1

1 **Explain the component modules of DBMS and their interaction, with the help of a diagram.** [10]

Explanation: 7 marks

Diagram: 3 marks

Solution:

DBMS Component Modules

• A higher-level stored data manager module of the DBMS controls access to DBMS information that is stored on disk.

• The DDL compiler processes schema definitions, specified in the DDL, and stores descriptions of the schemas (meta-data) in the DBMS catalog.

• The run-time database processor handles database accesses at run time.

The query compiler handles high-level queries that are entered interactively.

• The pre-compiler extracts DML commands from an application program written in a host programming language. These commands are sent to the DML compiler for compilation into object code for database access

Database System Utilities

1. Loading - loads existing data files (e.g., text files or sequential files) into the database.

2. Backup - this utility provides a backup copy of the database, usually by dumping the entire database onto tape.

3. File reorganization - can be used to reorganize a database file into a different file organization to improve performance.

4. Performance monitoring - monitors database usage and provides statics to the DBA.

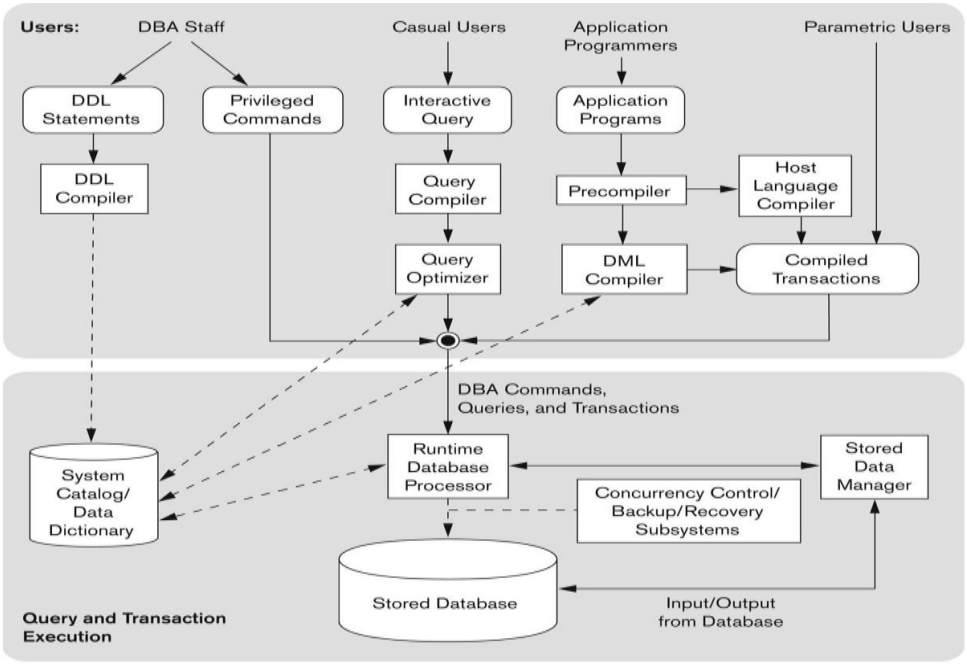


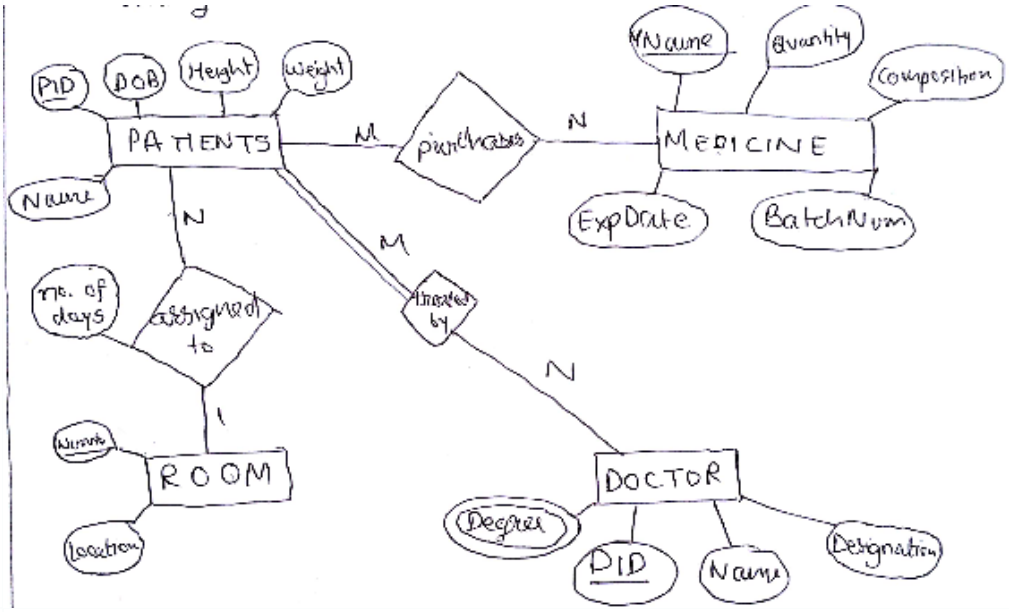
Figure 2.3
Component modules of a DBMS and their interactions.

2 Write an ER diagram for hospital management considering at least four entities. [10]

ER Diagram: 7 marks

Assumptions: 3 marks

Solutions:



3 Explain the syntax of a SELECT, INSERT, DELETE, UPDATE, ALTER statement in SQL With example. [10]

Select – 2 marks

Insert – 2 marks

Delete – 2 marks

Update – 2 marks

Alter – 2 makrs

Solutions:

DELETE Command

The DELETE command removes tuples from a relation. It includes a WHERE clause, similar to that used in an SQL query, to select the tuples to be deleted

DELETE FROM table_name WHERE condition;

DELETE FROM EMPLOYEE WHERE Lname='Brown';

The INSERT Command

In its simplest form, INSERT is used to add a single tuple to a relation. We must specify the relation name and a list of values for the tuple.

INSERT INTO EMPLOYEE(col1_name, col2_name, .., colN_name) VALUES (col1_value,col2_value, … , colN_value);

INSERT INTO EMPLOYEE VALUES(col1_value,col2_value, … , colN_value);

INSERT INTO EMPLOYEE VALUES ('Richard', 'K', 'Marini', '653298653', '1962-12-30', '98 Oak Forest, Katy, TX', 'M', 37000, '653298653', 4);

UPDATE Command

The **UPDATE** command is used to modify attribute values of one or more selected tuples. As in the DELETE command, a WHERE clause in the UPDATE command selects the tuples to be modified from a single relation.

UPDATE table_name SET column1=value, column2=value2,...WHERE condition

UPDATE PROJECT SET Plocation = 'Bellaire', Dnum = 5 WHERE Pnumber=10;

ALTER Command

The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.

Syntax:

- To add a column in a table, use the following syntax:

ALTER TABLE table_name ADD column_name datatype;

- To delete a column in a table, use the following syntax:

ALTER TABLE table_name DROP COLUMN column_name;

- To change the data type of a column in a table, use the following syntax:

ALTER TABLE table_name MODIFY COLUMN column_name datatype;

ALTER TABLE EMPLOYEE MODIFY DOB year;
ALTER TABLE EMPLOYEE DROP COLUMN DOB;

SELECT Statement:

The SELECT statement is used to select data from tables i.e. to query the database and retrieve selected data that match the criteria that you specify.

SELECT column_list FROM table-name
[WHERE Clause]
[GROUP BY clause]
[HAVING clause]
[ORDER BY clause];

SELECT * FROM EMPLOYEE;
SELECT EmpId,EmpName,EmpPosition FROM EMPLOYEE;
SELECT * FROM EMPLOYEE WHERE Salary>40000;
SELECT EmpPosition, Sum(Salary) FROM EMPLOYEE GROUP BY EmpPosition

4 Discuss the following (Concept, example and ER notation): [10]

i) Weak entity & Identifying relationship – 3 marks

Entity types that do not have key attributes of their own are called weak entity types. Entities belonging to a weak entity type are identified by being related to specific entities from another entity type in combination with some of their attribute values. We call this other entity type the **identifying or owner entity type** and we call the relationship type that relates a weak entity type to its owner the **identifying relationship** of the weak entity type. A weak entity type always has a total participation constraint (existence dependency) with respect to its identifying relationship, because a weak entity cannot be identified without an owner entity.

Consider the entity type DEPENDENT, related to EMPLOYEE, which is used to keep track of the dependents of each employee. The attributes of DEPENDENT are Name (the first name of the dependent), BirthDate, Sex, and Relationship (to the employee). Two dependents of two distinct employees may, by chance, have the same values for Name, BirthDate, Sex, and Relationship, but they are still distinct entities. They are identified as distinct entities only after determining the particular employee entity to which each dependent is related. Each employee entity is said to own the dependent entities that are related to it.

In ER diagrams, both a weak entity type and its identifying relationship are distinguished by surrounding their boxes and diamonds with double lines. A partial key, which is the set of attributes, is used to uniquely identify weak entities that are related to the same owner entity. The partial key attribute is underlined with a dashed or dotted line.

Weak Entity

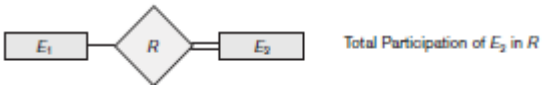
Indentifying Relationship

ii) Participation Constraints – 2 marks

The participation constraint specifies whether the existence of an entity depends on its being related to another entity via the relationship type. There are two types of participation constraints—total and partial.

If a company policy states that every employee must work for a department, then an employee entity can exist only if it participates in a WORKS_FOR relationship instance. Thus, the participation of EMPLOYEE in WORKS_FOR is called total participation, meaning that every entity in "the total set" of employee entities must be related to a department entity via WORKS_FOR. Total participation is also called existence dependency.

On the other hand, we do not expect every employee to manage a department, so the participation of EMPLOYEE in the MANAGES relationship type is partial, meaning that some or "part of the set of" employee entities are related to a department entity via MANAGES, but not necessarily all.



iii) Recursive Relationships and Role names. – 3 marks

Each entity type that participates in a relationship type plays a particular role in the relationship. The role name signifies the role that a participating entity from the entity type plays in each relationship instance, and helps to explain what the relationship means. For example, in the WORKS_FOR relationship type, EMPLOYEE plays the role of employee or worker and DEPARTMENT plays the role of department or employer.

In some cases the same entity type participates more than once in a relationship type in different roles. In such cases the role name becomes essential for distinguishing the meaning of the role that each participating entity plays. Such relationship types are called recursive relationships.

The SUPERVISION relationship type relates an employee to a supervisor, where both employee and supervisor entities are members of the same EMPLOYEE entity set. Hence, the EMPLOYEE entity type participates twice in SUPERVISION: once in the role of supervisor (or boss), and once in the role of supervisee (or subordinate).

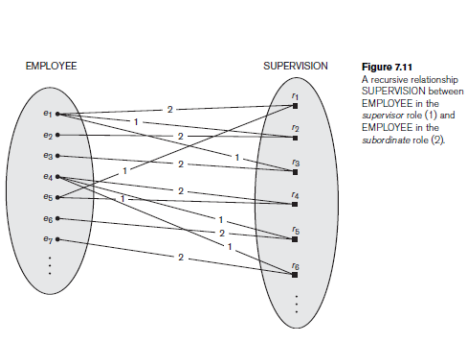
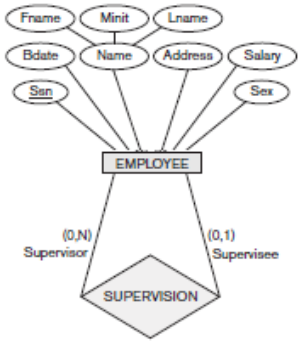


Figure 7.11
A recursive relationship SUPERVISION between EMPLOYEE in the *supervisor* role (1) and EMPLOYEE in the *subordinate* role (2).



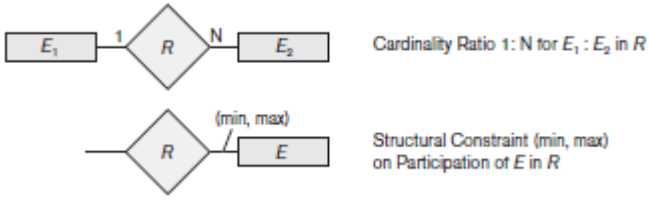
iv) Cardinality Ratio – 2 makrs

The cardinality ratio for a binary relationship specifies the maximum number of relationship instances that an entity can participate in.

For example, consider a binary relationship type WORKS_FOR between Department and Employee entity types, DEPARTMENT:EMPLOYEE is of cardinality ratio 1:N, meaning that each department can be related to numerous employees, but an employee can be related to (work for) only one department.

The possible cardinality ratios for binary relationship types are 1:1, 1:N, N:1, and M:N. The binary relationship MANAGES which relates a department entity to the employee who manages that department; the cardinality ratio is 1:1. This

represents the constraint that an employee can manage only one department and that a department has only one manager.
The relationship type WORKS_ON between Employee entity and the Project entity that he works for, is of cardinality ratio M:N, representing that an employee can work on several projects and a project can have several employees.



5 (a) List operation of relational algebra and explain the purpose of each with examples. [06]

Any 3 operations with explanation and examples – 3*2 marks

Solutions:

The following are the set theoretic operations are used to merge the elements of two sets in various ways in relational algebra,

- UNION
- INTERSECTION
- SET DIFFERENCE

When these operations are adapted to relational databases, the two relations on which any of the above three operations are applied must have the same type of tuples; this condition is called union compatibility.

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.

Example:

R:

Name
Ram
Raju
Rakesh

S:

Name
Ram
Rajesh
Ramu

R ∪ S:

Name
Ram
Raju
Rakesh
Rajesh
Ramu

INTERSECTION: The result of this operation, denoted by R ∩ S, is a relation that includes all tuples that are in both R and S.

Example:

R:

Name
Ram
Raju
Rakesh

S:

Name
Ram
Rajesh
Ramu

R ∩ S:

Name
Ram

SET DIFFERENCE: The result of this operation, denoted by R - S, is a relation that includes all tuples that are in R but not in S.

Example:

R:

Name
Ram
Raju
Rakesh

S:

Name
Ram
Rajesh
Ramu

R−S:

Name
Raju
Rakesh

(b) Differentiate between Strong entity set and weak entity set. [04]

Strong Entity: 2 marks

Weak Entity: 2 marks

Solutions:

Strong Entity:

Entity types that have key attributes of their own are called strong entity types. There entities need not to be associated with any other entity in order to be uniquely identified. Example: EMPLYEE Entity type with SSN as key attribute.

Weak entity:

Entity types that do not have key attributes of their own are called weak entity types. Entities belonging to a weak entity type are identified by being related to specific entities from another entity type in combination with some of their attribute values. We call this other entity type the identifying or owner entity type.

Example: DEPENDENT entity with no attribute or set of attributes unique in order to be made as key attribute. Hence Dependent is weak entity associated with Employee entity.

6 (a) Define an entity and an attribute. Explain the different types of attributes that occur in an ER model, with an example. [06]

Entity and Attribute definitions – 2 marks

Attribute types: 2 marks

Examples: 2 marks

Solutions:

Entities and Attributes

The basic object that the ER model represents is an entity, which is a thing in the real world with an independent existence. An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course).

Attribute—the particular properties that describe an entity. For example, an EMPLOYEE entity may be described by the employee's name, age, address, salary, and job.

Several types of attributes occur in the ER model: simple versus composite, single valued versus multivalued, and stored versus derived.

Composite versus Simple (Atomic) Attributes: Composite attributes can be divided into smaller subparts, which represent more basic attributes with independent meanings. For example, the Address attribute of the EMPLOYEE entity can be subdivided into Street_address, City, State, and Zip,3 with the values '2311 Kirby', 'Houston', 'Texas', and '77001.' Attributes that are not divisible are called simple or atomic attributes. Composite attributes can form a hierarchy; for example, Street_address can be further subdivided into three simple component attributes: Number, Street, and Apartment_number,

Single-Valued versus Multivalued Attributes: Most attributes have a single value for a particular entity; such attributes are called single-valued. For example, Age is a single-valued attribute of a person. one person may not have a college degree, another person may have one, and a third person may have two or more degrees; therefore, different people can have different numbers of values for the College_degrees attribute. Such attributes are called multivalued. A multivalued attribute may have lower and upper bounds to constrain the number of values allowed for each individual entity.

Stored versus Derived Attributes: In some cases, two (or more) attribute values are related—for example, the Age and Birth_date attributes of a person. For a particular person entity, the value of Age can be determined from the current (today's) date and the value of that person's Birth_date. The Age attribute is hence called a derived attribute and is said to be derivable from the Birth_date attribute, which is called a stored attribute.

(b) Discuss the main characteristics of database approach. How it differ from traditional file system. [04]

Each Characteristics: 1 marks

Solution:

- **Self-describing nature of a database system**: A fundamental characteristic of the database approach is that the database system contains not only the database itself but also a complete definition or description of the database structure and constraints. This definition is stored in the DBMS catalog, which contains information such as the structure of each file, the type and storage format of each data item, and various constraints on the data. The information stored in the catalog is called **meta-data**.
- **Support of multiple views of the data:** A database typically has many users, each of whom may require a different perspective or view of the database. A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored. Some users may not need to be aware of whether the data they refer to is stored or derived. A multiuser DBMS whose users have a variety of distinct applications must provide facilities for defining multiple views.
- **Insulation between programs and data, and data abstraction:** program-data independence, program-operation independence, data abstraction

  Program-data independence

  In traditional file processing, the structure of data files is embedded in the application programs, so any changes to the structure of a file may require *changing all programs* that access that file. By contrast, DBMS access programs do not require such changes in most cases. The structure of data files is stored in the DBMS catalog separately from the access programs. We call this property **program-data independence**.

  Program-operation independence

  In some types of database systems, such as object-oriented and object-relational systems ,users can define operations on data as part of the database definitions .User application programs can operate on the data by invoking these operations through their names and arguments, regardless of how the operations are implemented. This may be termed **program-operation independence**.

  Data abstraction

  The major purpose of a database system is to provide users with an **abstract view** of the system. The system hides certain details of how data is stored and created and maintained. Complexity should be hidden from database users. It gives an architecture is to separate the user applications and the physical database.

  - ***Levels of Abstraction***

    – Physical schema Defines how data is stored

    – Conceptual schema or logical schema Defines data in terms of a data model

    – External schema or view level Defines a number of simplified domain-specific views

- **Sharing of data and multiuser transaction processing:** A multiuser DBMS, as its name implies, must allow multiple users to access the database at the same time. This is essential if data for multiple applications is to be integrated and maintained in a single database.