CMR
INSTITUTE OF
TECHNOLOGY

MCA- Internal Assesment Test – I – Answer Key

Sub:	**Database Management System Sem: 2**	**Code:**	**16MCA23**

Answer Any FIVE FULL Questions

1(a) Explain the characteristics of database approach. (4)

1. Self-describing nature of a database system.
2. Insulation between programs and data, and data abstraction.
3. Support of multiple views of the data.
4. Sharing of data and multiuser transaction processing.

(b) Discuss the classification of Database Management System. (6)

1. **Centralized database system:** the DBMS and database are stored at a single site that is used by several other systems too
2. **Distributed database system**: the actual database and the DBMS software are distributed from various sites that are connected by a computer network
3. **Heterogeneous distributed database system**: different sites might use different DBMS software, but there is additional common software to support data exchange between these sites
4. **Homogeneous distributed database systems**: use the same DBMS software at multiple sites
5. **Multiuser database system**: a database management system which supports multiple users concurrently
6. **Object-oriented data model**: a database management system in which information is represented in the form of objects as used in object-oriented programming
7. **Single-user database system**: a database management system which supports one user at a time
8. **Traditional models**: data models that preceded the relational model

2 (a) Define data independence? Explain the various types of data independence.

A major objective for three-level architecture is to provide data independence, which means that upper levels are unaffected by changes in lower levels.

There are two kinds of data independence: **(2)**

- Logical data independence
- Physical data independence

Logical Data Independence: (2)

i. Logical data independence indicates that the conceptual schema can be changed without affecting the existing external schemas. The change would be absorbed by the mapping between the external and conceptual levels. Logical data independence also insulates application programs from operations such as combining two records into one or splitting an existing record into two or more records. This would require a. change in the external/conceptual mapping so as to leave the external view unchanged.

Physical Data Independence: (2)

ii. Physical data independence indicates that the physical storage structures or devices could be changed without affecting conceptual schema. The change would be absorbed by the mapping between the conceptual and internal levels. Physic 1data independence is achieved by the presence of the internal level of the database and the n, lPping or transformation from the conceptual level of the database to the internal level. Conceptual level to internal level mapping, therefore provides a means to go from the conceptual view (conceptual records) to the internal view and hence to the stored data in the database (physical records).

(b) Explain DBMS interfaces (4)

Menu based
Form based
GUI
Natural Language Interface
Speech I/O
Interfaces Parametric users
Interface for DBA

3(a) **Explain the database System environment (5)**

DBMS Component modules
Database System Utilities
Tools, application environments and communication facilities
Application development environments

(b) **Define the following terms, with an example with respect to ER model:**

Entity Set, Relationship type, Cardinality ratio, Participation, Weak entity set. **(5)**

Entity Set:
An entity is an object that exists and is distinguishable from other objects. For instance, John Harris with S.I.N. 890-12-3456 is an entity, as he can be uniquely identified as one particular person in the universe.
An entity may be concrete (a person or a book, for example) or abstract (like a holiday or a concept). An entity set is a set of entities of the same type (e.g., all persons having an account at a bank).

Relationship type:
A relationship is any association,linkage, or connection between the entities of interest to the business; it is a two directional, significant association between two entities, or between an entity and itself

Cardinality ratio:

In database design, the cardinality or fundamental principle of one data table with respect to another is a critical aspect. The relationship of one to the other must be precise and exact between each other in order to explain how each table links together.

Participation:

A participation constraint defines the number of times an object in an object class can participate in a connected relationship set. Every connection of a relationship set must have a participation constraint. However, participation constraints do not apply to relationships.

Weak entity set:

An entity set that does not have a primary key is referred to as a weak entity set. The existence of a weak entity set depends on the existence of a strong entity set; it must relate to the strong set via a one-to-many relationship set.

4(a) Explain the different types of attributes, which occur in ER model (10)

- simple/atomic vs. composite
- single-valued vs. multi-valued (or set-valued)
- stored vs. derived (Note from instructor: this seems like an implementation detail that ought not be considered at this (high) level of abstraction.)

A composite attribute is one that is composed of smaller parts. An atomic attribute is indivisible or indecomposable.

Example 1: A BirthDate attribute can be viewed as being composed of (sub-)attributes month, day, and year (each of which would probably be viewed as being atomic).

To describe the structure of a composite attribute, one can draw a tree (as in the aforementioned Figure 7.4). In case we are limited to using text, it is customary to write its name followed by a parenthesized list of its sub-attributes. For the examples mentioned above, we would write

BirthDate(Month, Day, Year)
Address(StreetAddr(StrNum, StrName, AptNum), City, State, Zip)

Single- vs. multi-valued attribute: Consider a PERSON entity. The person it represents has (one) SSN, (one) date of birth, (one, although composite) name, etc. But that person may have zero or more academic degrees, dependents, or (if the person is a male living in Utah) spouses! How can we model this via attributes Academic Degrees, Dependents, and Spouses? One way is to allow such attributes to bemulti-valued (perhaps set-valued is a better term), which is to say that we assign to them a (possibly empty) set of values rather than a single value.

To distinguish a multi-valued attribute from a single-valued one, it is customary to enclose the former within curly braces (which makes sense, as such an attribute has a value that is a set, and curly braces are traditionally used to denote sets). Using the PERSON example from above, we would depict its structure in text as

PERSON(SSN, Name, BirthDate(Month, Day, Year), { AcademicDegrees(School, Level, Year) }, { Dependents }, ...)

A more complicated example of a complex attribute is AddressPhone . Its structure is given by

{ AddressPhone({ Phone(AreaCode, Number) }, Address(StrAddr(StrNum, StrName, AptNum), City, State, Zip)) }

5(a) **Summarize the different integrity constraints and give examples for Check and UNIQUE constraint. (4)**

Integrity constraints provide a way of ensuring that changes made to the database by authorized users do not result in a loss of data consistency. We saw a form of integrity constraint with E-R models: key declarations: stipulation that certain attributes form a candidate key for the entity set.

CHECK constraints let you enforce very specific integrity rules by specifying a check condition. The condition of a CHECK constraint has some limitations:
It must be a Boolean expression evaluated using the values in the row being inserted or updated, and
It cannot contain sub queries; sequences; the SQL functions SYSDATE, UID, USER, or USERENV; or the pseudo columns LEVEL or ROWNUM.

UNIQUE constraint: The UNIQUE constraint ensures that all values in a column are different. Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns. A PRIMARY KEY constraint automatically has a UNIQUE constraint. However, you can have many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.

```
CREATE TABLE Persons (
    ID int NOT NULL UNIQUE,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int
);
```

(b) Explain authorization in SQL. (6)

A privilege is a right to execute a particular type of SQL statement or to access another user's object.

Some examples of privileges include the right to:

- Connect to the database (create a session)
- Create a table
- Select rows from another user's table
- Execute another user's stored procedure

we grant privileges to users so these users can accomplish tasks required for their jobs. We should grant a privilege only to a user who requires that privilege to accomplish the necessary work. Excessive granting of unnecessary privileges can compromise security. A user can receive a privilege in two different ways:

- We can grant privileges to users explicitly. For example, you can explicitly grant to user SCOTT the privilege to insert records into the employees table.
- We can also grant privileges to a role (a named group of privileges), and then grant the role to one or more users. For example, we can grant the privileges to select, insert, update, and delete records from the employees table to the role named clerk, which in turn we can grant to users scott and brian.

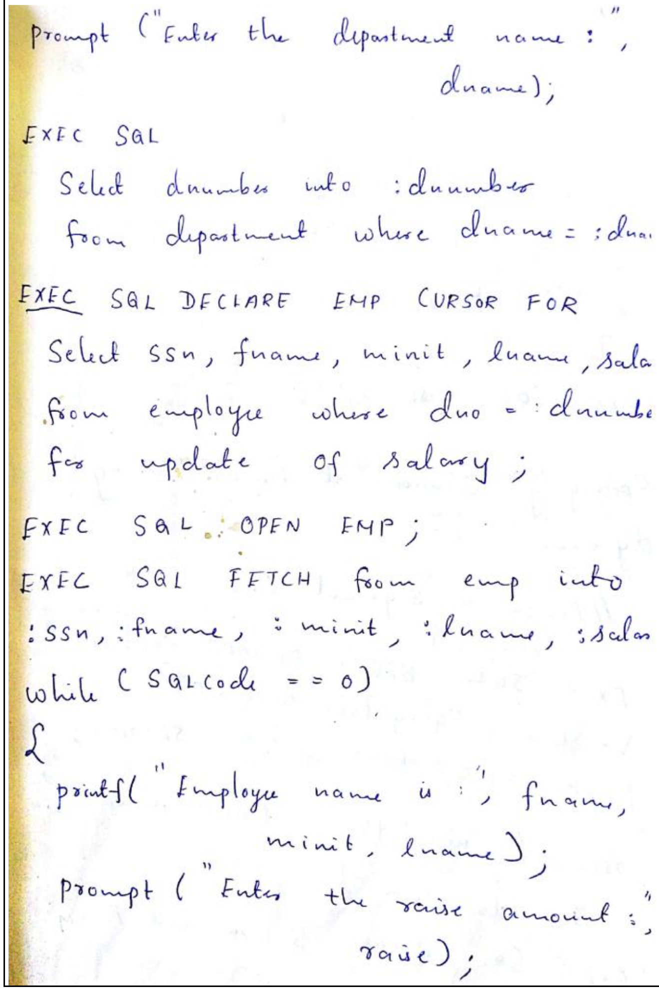
Because roles allow for easier and better management of privileges, you should normally grant privileges to roles and not to specific users.

Grant <privilege list> on <relation-name> or <view-name. to ,user/role list>

Grant (select/insert/update) on <table-name. to user1, user2;

6(a) Explain with code how to retrieve multiple tuples in Embedded SQL. **(10)**

Specify the cursor using a DECLARE CURSOR statement.
Perform the query and build the result table using the OPEN statement.
Retrieve rows one at a time using the FETCH statement.
Process rows with the DELETE or UPDATE statements (if required).
Terminate the cursor using the CLOSE statement.



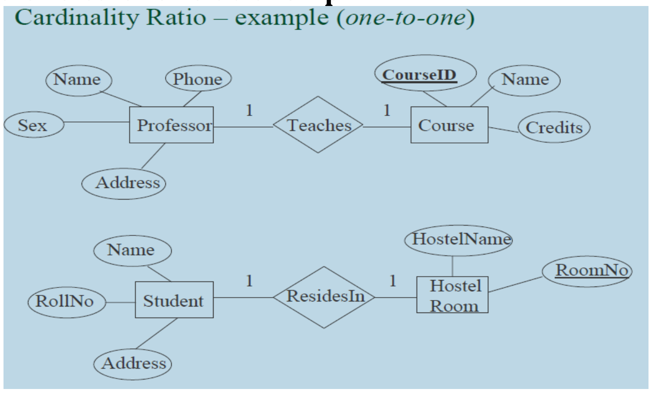
```
Prompt ("Enter the department name :",
                         dname);
EXEC SQL
  Select dnumber into :dnumber
  from department where dname = :dna
EXEC SQL DECLARE EMP CURSOR FOR
  Select ssn, fname, minit, lname, sala
  from employee where dno = :dnumbe
  for update of salary ;
EXEC SQL : OPEN EMP ;
EXEC SQL FETCH from emp into
:ssn, :fname, : minit, :lname, :salar
while ( SQLcode == 0)
{
  printf( "Employee name is :", fname,
               minit, lname);
  Prompt ( "Enter the raise amount :",
                  raise);
EXEC SQL
  update EMPLOYEE set Salary = Salar
                                :raise
  where Current of EMP;

  EXEC SQL FETCH from Emp into
    :ssn, :fname, :minit, :lname,
    :salary;
 }
EXEC SQL CLOSE EMP;
```

Specifying Queries at Runtime using dynamic SQL:

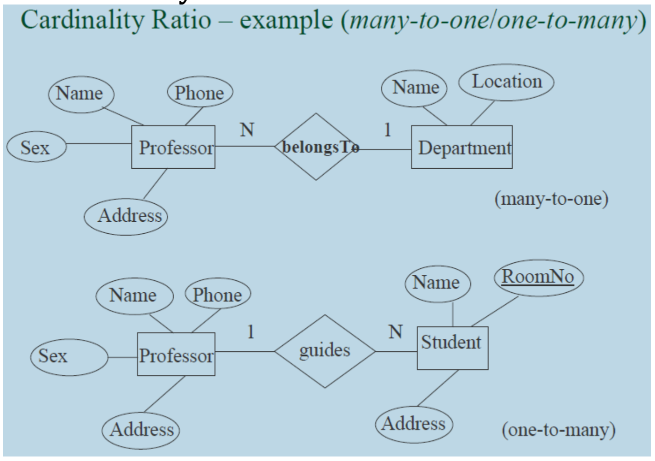
```
//Program Segment E3:

EXEC SQL BEGIN DECLARE SECTION;
Varchar sqlupdatestring [256];
EXEC SQL END DECLARE SECTION;
...
prompt ("Enter the update command",
   sqlupdatestring);
EXEC SQL PREPARE Sqlcommand FROM
          :sqlupdatestring;
EXEC SQL EXECUTE Sql
```

7(a) Explain at least one example of your environment (real world) for each of the following relationships: (4)

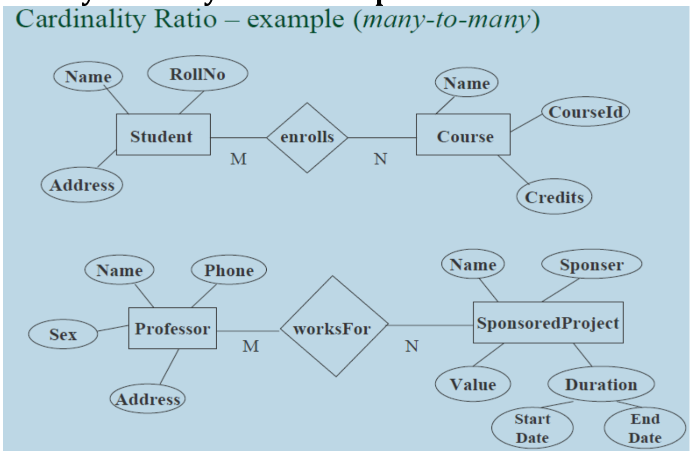
(a) a one-to-one relationship



(b) a one-to-many relations



(c) a many-to-many relationship



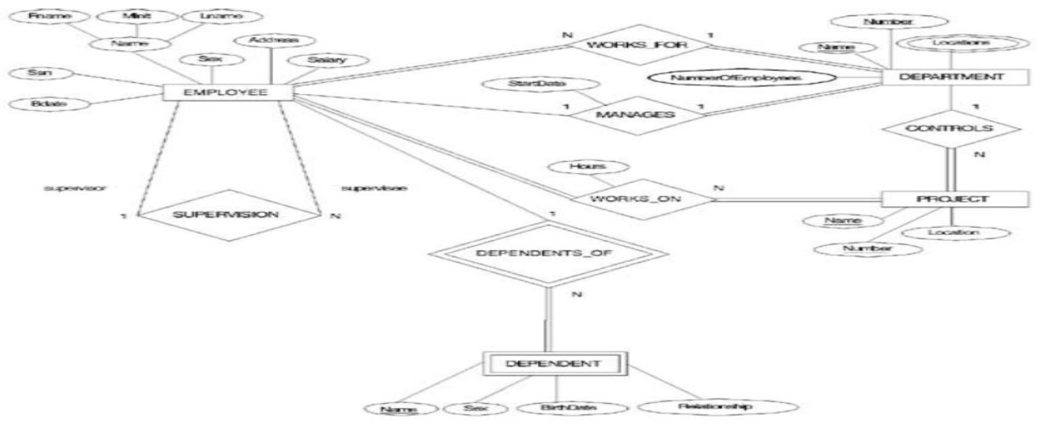
8(a) Consider the following structure of a database that keeps employees details and complete the queries given:

Name	Null?	Type
EMPID	NOT NULL	NUMBER(5)
FNAME		VARCHAR2(25)
LNAME		VARCHAR2(25)
EMAIL		VARCHAR2(20)
PHONE		NUMBER(10)
DOJ		DATE
JOBID		VARCHAR2(4)
SAL		NUMBER(8)
MGRID		NUMBER(3)
DID		NUMBER(3)

(10)

1.Write a query to get unique department ID from employee table
select *from gemp where did in (select did from gemp group by did having count(*)=1);
2. Write a query to get all employee details from the employee table order by first name, descending
select *from gemp order by fname desc;
3.Write a query to get the employee ID, names (first_name, last_name), salary in ascending order of salary.
select empid,fname,lname,sal from gemp order by sal asc;
4.Write a query to get the number of jobs available in the employee table.
select count(distinct(jobid)) "Number of Jobs" from gemp;
5.Write a query to select fname having ma.
select * from gemp where fname like '%ma%' ;

(b) Summarize the company database with a brief explanation and ER diagram. (6)


Fname
Minit
Lname
Name
Address
Sex
Salary
Ssn
Bdate
EMPLOYEE
N
WORKS_FOR
1
Number
Name
Locations
DEPARTMENT
NumberOfEmployees
StartDate
1
MANAGES
1
1
CONTROLS
N
supervisor
supervisee
1
SUPERVISION
N
Hours
WORKS_ON
N
PROJECT
Name
Location
Number
1
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
BirthDate
Relationship