ABES Engineering College, Ghaziabad

B.Tech Odd Semester Sessional ST-1

DBMS KCS 501 (Odd Semester 2022-23)

Date of Exam: 17.10.2022

Solution

Section A

1 a) A relation is given **(Registration_Num, UID, BankAccount_Num, Name, Hostel_Room)**.

Now, **Registration_Num** is unique for each student. So with this, we can identify each student.

[1 mark]

Hence, this can be the primary key. **UID**: It's an identification number for a person in a country. (Say you're in India and your **UID** is 0243. Someone in Pakistan may also have the same **UID** as 0243). So, if all students are from India (that is, the same country) then their **UID** will be different and then **UID** will be a Candidate key. **BankAccount_Num** is not a candidate key, because a student can have multiple accounts or joint accounts. We cannot identify each student uniquely with **BankAccount_Num**. **[1 Mark]**

1b) Strong entities E1 and E2 are represented as separate tables.

In addition to that many-to-many relationships (R2) must be converted as separate table by having primary keys of E1 and E2 as foreign keys. **[1 Mark]**

One-to-many relationship (R1) must be transferred to 'many' side table(i.e. E2) by having primary key of one side(E1) as foreign key(this way we need not to make a separate table for R1).

[1 Mark]

Hence we will have minimum of 3 tables.

1c) Examples of non-candidate keys include the following: {name}, {age}.
(Note that {gpa} can not be declared as a non-candidate key from this evidence alone even though common sense tells us that clearly more than one student could have the same grade point average.) **[2 Marks]**

1d) The integrity constraint is that the GPA stored in the database is the GPA of the course grades stored. **[1 Mark]**

That constraint could be violated if a transaction updated a course grade and aborted before it could update the GPA. **[1 Mark]**

1e) Apply membership test for all the given Functional Dependencies.

1. $CD \rightarrow AC$

   $CD^+ = CDEAB$

2. $BD \rightarrow CD$

   $BD^+ = BD$

i.e. BD cannot derive CD and hence is not implied **[1 Mark]**

Repeat the same process

CD→ AC

BC→ CD

AC→ BC

Are the FDS which can be implied from above FDs **[1 Mark]**

Section B

2 a)

1. **Database Administrator (DBA)** :
   Database Administrator (DBA) is a person/team who defines the schema and also controls the 3 levels of database.
   The DBA will then create a new account id and password for the user if he/she need to access the data base.
   DBA is also responsible for providing security to the data base and he allows only the authorized users to access/modify the data base**[1 Mark]**
2. **Naive / Parametric End Users** :
   Parametric End Users are the unsophisticated who don't have any DBMS knowledge but they frequently use the data base applications in their daily life to get the desired results. For examples, Railway's ticket booking users are naive users. Clerks in any bank is a naive user because they don't have any DBMS knowledge but they still use the database and perform their given task. **[1 Mark]**
3. **System Analyst :**
   System Analyst is a user who analyzes the requirements of parametric end users. They check whether all the requirements of end users are satisfied**[1 Mark]**
4. **Sophisticated Users :**
   Sophisticated users can be engineers, scientists, business analyst, who are familiar with the database. They can develop their own data base applications according to their requirement. They don't write the program code but they interact the data base by writing SQL queries directly through the query processor. **[1 Mark]**
5. **Data Base Designers :**
   Data Base Designers are the users who design the structure of data base which includes tables, indexes, views, constraints, triggers, stored procedures. He/she controls what data must be stored and how the data items to be related.
6. **Application Programmers :**
   Application Program are the back end programmers who writes the code for the application programs.They are the computer professionals. These programs could be written in Programming languages such as Visual Basic, Developer, C, FORTRAN, COBOL etc.
7. **Casual Users / Temporary Users :**
   Casual Users are the users who occasionally use/access the data base but each time when they access the data base they require the new information, for example, Middle or higher level manager. **[1 Mark]**

b)

1. **Primary Key**

The primary key refers to a column or a set of columns of a table that helps us identify all the records uniquely present in that table. A table can consist of just one primary key. Also, this primary key cannot consist of the same values reappearing/repeating for any of its rows. All the values of a primary key have to be different, and there should be no repetitions.

The PK (PRIMARY KEY) constraint that we put on a column/set of columns won't allow these to have a null value or a duplicate. Any table can consist of only a single primary key constraint. A foreign key that refers to it can never change the values present in the primary key. **[1 Mark]**

2. **Super Key**

A super key refers to the set of all those keys that help us uniquely identify all the rows present in a table. It means that all of these columns present in a table that can identify the columns of that table uniquely act as the super keys.

A super key is a candidate key's superset . We need to pick the primary key of any table from the super key's set so as to make it the table's identity attribute**[1 Mark]**

3. **Candidate Key**

The candidate keys refer to those attributes that identify rows uniquely in a table. In a table, we select the primary key from a candidate key. In a table, there can be multiple candidate keys.

Candidate keys are a subset of Super keys. They contain only those attributes which are required to uniquely identify tuples.
All Candidate keys are Super keys. But the vice-versa is not true. **[1 Mark]**

4. **Surrogate Key**

Surrogate key also called a synthetic primary key, is generated when a new record is inserted into a table automatically by a database that can be declared as the primary key of that table. It is the sequential number outside of the database that is made available to the user and the application or it acts as an object that is present in the database but is not visible to the user or application.

[1 Mark]

5. **Foreign Key**

We use a foreign key to establish relationships between two available tables. The foreign key would require every value present in a column/set of columns to match the referential table's primary key. A foreign key helps us to maintain data as well as referential integrity.Foreign keys may accept non-unique and null values. **[1 Mark]**

3) a

Data Model gives us an idea that how the final system will look like after its complete implementation. It defines the data elements and the relationships between the data elements. Data Models are used to show how data is stored, connected, accessed and updated in the database management system.

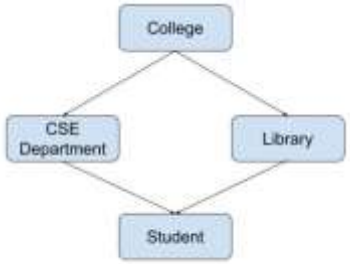
Hierarchical Model

Hierarchical Model was the first DBMS model. This model organises the data in the hierarchical tree structure. The hierarchy starts from the root which has root data and then it expands in the form of a tree adding child node to the parent node. This model easily represents some of the real-world relationships like food recipes, sitemap of a website etc. Example: We can represent the relationship between the shoes present on a shopping website in the following way **[1 Mark]**

Hierarchical Model

Network Model

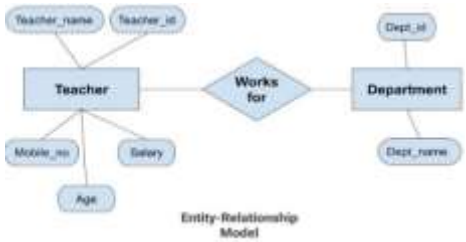
This model is an extension of the hierarchical model. It was the most popular model before the relational model. This model is the same as the hierarchical model, the only difference is that a record can have more than one parent. It replaces the hierarchical tree with a graph. Example: In the example below we can see that node student has two parents i.e. CSE Department and Library. This was earlier not possible in the hierarchical model. **[1 Mark]**



Network Model

Entity-Relationship Model

Entity-Relationship Model or simply ER Model is a high-level data model diagram. In this model, we represent the real-world problem in the pictorial form to make it easy for the stakeholders to understand. It is also very easy for the developers to understand the system by just looking at the ER diagram. We use the ER diagram as a visual tool to represent an ER Model **[1 Mark]**



Entity-Relationship Model

Relational Model

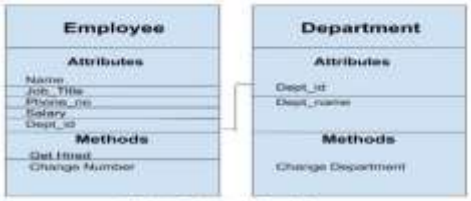
Relational Model is the most widely used model. In this model, the data is maintained in the form of a two-dimensional table. All the information is stored in the form of row and columns. The basic structure of a relational model is tables. So, the tables are also called relations in the relational model. Example: In this example, we have an Employee table. **[1 Mark]**

Emp_id	Emp_name	Job_name	Salary	Mobile_no	Dep_id	Project_id
AfterA001	John	Engineer	100000	9111037890	2	99
AfterA002	Adam	Analyst	50000	9587569214	3	100
AfterA003	Kande	Manager	850000	7895212355	2	65

EMPLOYEE TABLE

Object-Oriented Data Model

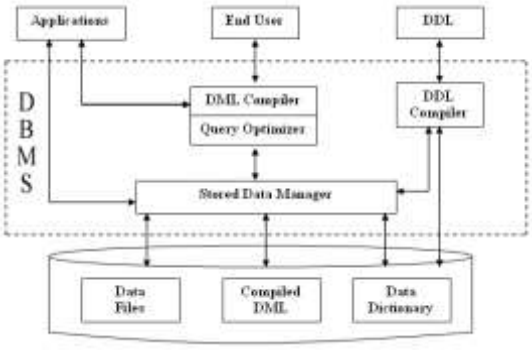
The real-world problems are more closely represented through the object-oriented data model. In this model, both the data and relationship are present in a single structure known as an object. We can store audio, video, images, etc in the database which was not possible in the relational model(although you can store audio and video in relational database, it is adviced not to store in the relational database). In this model, two are more objects are connected through links. We use this link to relate one object to other objects. This can be understood by the example given below. **[1 Mark]**



Object_Oriented_Model

b)

DBMS (Database Management System) acts as an interface between the user and the database. The user requests the DBMS to perform various operations (insert, delete, update and retrieval) on the database. The components of DBMS perform these requested operations on the database and provide necessary data to the users. The various components of DBMS are shown below: -



[2 Marks]

1. DDL Compiler - Data Description Language compiler processes schema definitions specified in the DDL. It includes metadata information such as the name of the files, data items, storage details of each file, mapping information and constraints etc. **[1 Mark]**

2. DML Compiler and Query optimizer - The DML commands such as insert, update, delete, retrieve from the application program are sent to the DML compiler for compilation into object code for database access. The object code is then optimized in the best way to execute a query by the query optimizer and then send to the data manager. **[1 Mark]**

3. Data Manager - The Data Manager is the central software component of the DBMS also knows as Database Control System.

4. Data Dictionary - Data Dictionary is a repository of description of data in the database. It contains information about data names, constraints,relationships etc.

5. Data Files - It contains the data portion of the database.

6. Compiled DML - The DML complier converts the high level Queries into low level file access commands known as compiled DML.

7. End Users – Interacts with DBMS to fetch information. **[1 Mark]**

4a) Using definition of equivalence of FD set, let us determine the right-hand side of the FD set of P using FD set Q.
Given P = { X → Y, XY → Z, W → XZ, W → V} and Q = { X → YZ, W → XV }
Let's find closure of the left side of each FD of P using FD Q.

1. X+ = XYZ (using X → YZ)
2. XY+ = XYZ (using X → YZ)
3. W+ = WXVYZ (using W → XV and X → YZ)
4. W+ = WXVYZ (using W → XV and X → YZ) **[2 Marks]**

Now compare closure of each X, XY, W and W calculated using FD Q with the right-hand side of FD P. Closure of each X, XY, W and W has all the attributes which are on the right-hand side of each FD of P. Hence, we can say **P is a subset of Q----------1** **[1 Mark]**
→Using definition of equivalence of FD set, let us determine the right-hand side of the FD set of Q using FD set P.
Given P = { X → Y, XY → Z, W → XZ, W → V} and Q = { X → YZ, W → XV }
Let us find closure of the left side of each FD of Q using FD P.

1. X+ = XYZ (using X → Y and XY → Z)
2. W+ = WXZVY (using W → XZ, W → V, and X → Y)

Now compare closure of each X, W calculated using FD P with the right-hand side of FD Q. Closure of each X and W has all the attributes which are on the right-hand side of each FD of Q. Hence, we can say **Q is a subset of P----------2**
From 1 and 2 we can say that P = Q **[2 Marks]**

4b) From the formal Definition of Functional Dependency : Functional dependency in DBMS refers to a relationship that is present between attributes of any table that are dependent on each other. E. F. Codd introduced it, and it helps in avoiding data redundancy and getting to know more about bad designs.

For the figure below : -

X	Y	Z
x_1	y_1	z_1
x_1	y_1	z_2
x_2	y_1	z_1
x_2	y_1	z_3

As Three attributes are associated in Relation : -

The total different number of functional dependency that can exists are as follows:-

For X

ϕ→ ϕ (Does not add value). This FD is discarded

X→X (Trivial Dependency). This FD is discarded

X→y (Non-Trivial Dependency). This FD hold as per the data provided in figure. For every Unique value of X , same value is for Y.

X→Z (Non-Trivial Dependency). This FD will not hold as for same value of X there are different values of Z.

XY→Z (Non-Trivial Dependency). This FD will not hold as for same value of XY there are different values of Z.

XZ→Y (Non-Trivial Dependency). This FD will hold as for same value of XZ there are same values of Y.

For Y

ϕ→ ϕ (Does not add value). This FD is discarded

y→Y (Trivial Dependency). This FD is discarded

y→Z (Non-Trivial Dependency). This FD will not hold as for same value of Y there are different values of Z.

y→x (Non-Trivial Dependency). This FD will not hold as for same value of y there are different values of x.

XY→Z (Non-Trivial Dependency). This FD will not hold as for same value of XY there are different values of Z.

YZ→X (Non-Trivial Dependency). This FD will not hold as for same value of YZ there are different values of X.

For Z

ϕ→ ϕ (Does not add value). This FD is discarded

Z→z (Trivial Dependency). This FD is discarded

Z→y (Non-Trivial Dependency). This FD hold as per the data provided in figure. For every Unique value of Z, same value is for Y.

Z→x (Non-Trivial Dependency). This FD will not hold as for same value of Z there are different values of x.

XZ→Y (Non-Trivial Dependency). This FD hold as per the data provided in figure. For every Unique value of XZ, same value is for Y.

YZ→X (Non-Trivial Dependency). This FD will not hold as for same value of YZ there are different values of X.

Thus the FDs which hold in the above figure are : -

X→Y , **XZ→Y** , **Z→Y** **[2.5 Marks]**

b) Now if the value of attribute Z of the last record in the relation is changed from z3 to z2 **will repeat the above process again to find New FDs which hold**

For X

ϕ→ ϕ (Does not add value). This FD is discarded

X→X (Trivial Dependency). This FD is discarded

X→y (Non-Trivial Dependency). This FD hold as per the data provided in figure. For every Unique value of X , same value is for Y.

X→Z (Non-Trivial Dependency). This FD will not hold as for same value of X there are different values of Z.

XY→Z (Non-Trivial Dependency). This FD will not hold as for same value of XY there are different values of Z.

XZ→Y (Non-Trivial Dependency). This FD will hold as for same value of XZ there are same values of Y.

For Y

ϕ→ ϕ (Does not add value). This FD is discarded

y→Y (Trivial Dependency). This FD is discarded

y→Z (Non-Trivial Dependency). This FD will not hold as for same value of Y there are different values of Z.

y→x (Non-Trivial Dependency). This FD will not hold as for same value of y there are different values of x.

XY→Z (Non-Trivial Dependency). This FD will not hold as for same value of XY there are different values of Z.

YZ→X (Non-Trivial Dependency). This FD will not hold as for same value of YZ there are different values of X.

For Z

ϕ→ ϕ (Does not add value). This FD is discarded

Z→z (Trivial Dependency). This FD is discarded

Z→y (Non-Trivial Dependency). This FD hold as per the data provided in figure. For every Unique value of Z, same value is for Y.

Z→x (Non-Trivial Dependency). This FD will not hold as for same value of Z there are different values of x.

XZ→Y (Non-Trivial Dependency). This FD hold as per the data provided in figure. For every Unique value of XZ, same value is for Y.

YZ→X (Non-Trivial Dependency). This FD will not hold as for same value of YZ there are different values of X.

Thus the FDs which hold in the above figure are : -

X→Y , XZ→Y , Z→Y **[2.5 Marks]**

Section C

5 a) For creating of ER diagram following steps are followed: -

Finding of Strong Entities and Weak Entities with attributes :-

Strong Entities are

Student
Instructor
Course
Weak Entities are

Course offerings

Next step is finding the relationship and cardinalities between entities, which are listed below:-

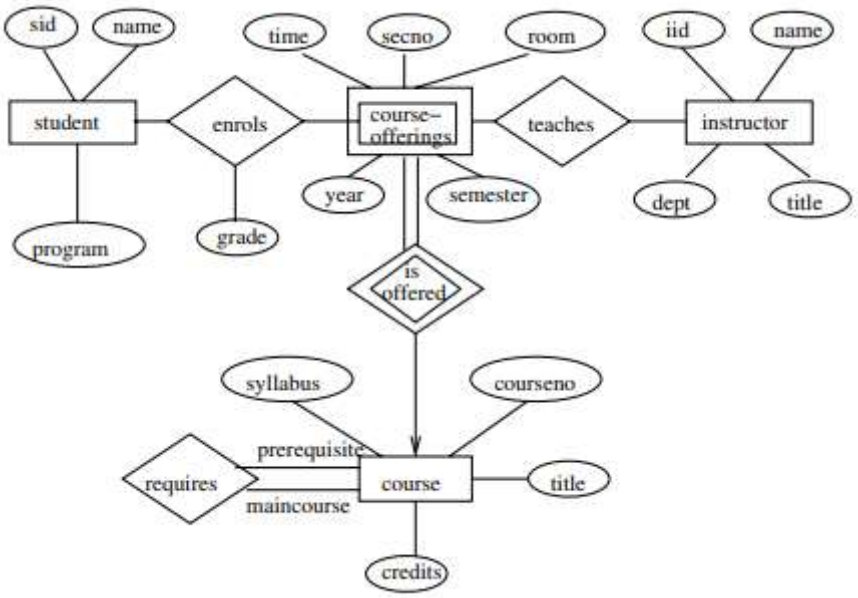
Enrols (between Student and Course Offerings with Many to Many Relationship (M:N))

Teaches(between Course Offerings and Instructor with Many to Many Relationship (M:N)))

Is Offered(between Course Offerings and Course with Many to Many Relationship (M:1)))

[3 Marks]

Now deploying all these points final ER Diagram is:-



[7 Marks]

5b i)

File System	DBMS
Used to manage and organise the files stored in the hard disk of the computer	A software to store and retrieve the user's data
Redundant data is present	No presence of redundant data
Query processing is not so efficient	Query processing is efficient
Data consistency is low	Due to the process of normalisation, the data consistency is high
Less complex, does not support complicated transactions	More complexity in managing the data, easier to implement complicated transactions
Less security	Supports more security mechanisms
Less expensive in comparison to DBMS	Higher cost than the File system
Does not support crash recovery	Crash recovery mechanism is highly supported

[5 Marks]

5b ii) Data Independence is the property of DBMS that helps you to change the Database schema at one level of a database system without requiring to change the schema at the next higher level.

Two levels of data independence are 1) Physical and 2) Logical

- Logical data is database data, which means it stores information about how data is managed within the database. Logical data independence is a method that makes sure that if we make modifications to the table format, the data should not be affected.
- The mapping between the external and conceptual levels will absorb any changes made.

In other words, to distinguish the external level from the conceptual view, logical data independence is used. Any modifications to the conceptual representation of the data will not affect the user's view of the data. **[2.5 Marks]**

Examples of Logical Data Independence:

- Without rewriting current application scripts, you can add, modify, or delete a new attribute, entity, or relationship.
- To divide an existing record into two or more records.
- Merging two records into a single one.

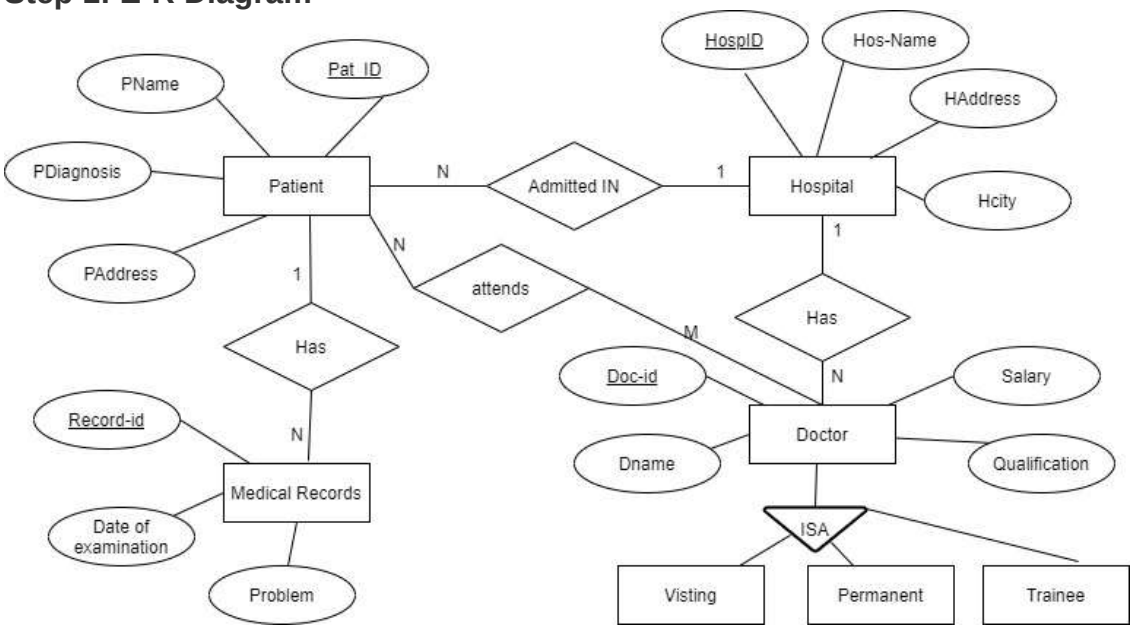
- Physical data independence allows you to distinguish between conceptual and internal/physical levels. It allows you to describe the database logically without needing to identify physical structures.
- Physical data independence allows you to modify physical storage structures or devices without affecting the conceptual model of the database. Any changes made at the internal level would be absorbed by the mapping between the conceptual and internal levels, preventing any modifications to the conceptual level.

Examples of Physical Data Independence:

- Changing from one data structure to another.
- Making use of new storage technology, such as a hard drive or magnetic tapes
- Change the location of the database from one drive to another.
- Changing the database's file organization. **[2.5 Marks]**

6 a)

Step 1: E-R Diagram



[4 Marks]

Converting the E-R Diagram into Tables

b. Converting entity to table and attribute to columns

Hospital

Hosp-id	**Primary Key**
HCity	
HAddress	
Hos-Name	
Pat-id	Foreign key references to Pat-id of Patient table
Doc-id	Foreign key references to Doc-id of Doctor table

Patient

Pat-id	**Primary Key**
PName	
PAddress	

Pat-id	**Primary Key**
PDiagnosis	
Record-id	Foreign key references to Record-id of Medical Record table
Hosp-id	Foreign key references to Hosp-id of Hospital table

Medical Record

Record-id	**Primary Key**
Problem	
Date_of_examination	
Pat-id	Foreign key references to Pat-id of Patient table

Doctor

Doc-id	**Primary Key**
DName	
Qualification	

Doc-id	**Primary Key**
Salary	
Hosp-id	Foreign key references to Hosp-id of Hospital table

Step 3: Mapping of Attributes

- **Simple Attributes**

  Simple attributes which cannot be divided into subparts.

  Example: Salary of Doctor

- **Composite Attributes**

  Composite Attributes which can be divided into subparts.

  Example: Patient Name, Doctor Name

Step 4: Mapping of Relationships

b. Foreign Key approach

Hosp_patient

Pat-id	**Hospital table makes foreign key references to Pat-id of Patient table**
Hosp-id	Patient table makes foreign key references to Hosp-id of Hospital table

Hosp_Doctor

Hosp-id	**Doctor table makes foreign key references to Hosp-id of Hospital table**
Doc-id	Hospital table makes foreign key references to Doc-id of Doctor table

PatiPPatient_MedicalRecord

Pat-id	**Medical Record table makes foreign key references to Pat-id of Patient table**
Record-id	Patient table makes foreign key references to Record-id of Medical Record table

Patient_Doctor

Pat-id	**Pat_id and Doctor_id will act as composite key for new table in relationship of Patient and Doctor**
Doctor-id	**Patient table makes foreign key references to Record-id of Medical Record table**

→Considering only the superclass is mapped to table, which will remove the null values.

Step 5: Identifying the relationships

a. Hospital has a set of patients.

Therefore the relations is 1……..N.

b. Hospital has a set of doctors.

Therefore the relations is 1……..N.

c. Doctor are associated with each patient.

Therefore the relations is N……..1.

d. Each patient has record of various test and examination conducted.

Therefore the relations is 1……..N. **[3 Marks]**

SQL : DDL Statement

```
CREATE
TABLE
Patient(
        Pat_id varchar(50) PRIMARY KEY,
        PName varchar(30) NOT NULL,
        PDiagnosis varchar(50) NOT NULL,
        Paddress varchar(60) NOT NULL,
        FOREIGN KEY (HospID) REFERENCES Hospital(HospID)
        );

CREATE
TABLE
Hospital(
         Hospid varchar(50) PRIMARY KEY,
         Hosname varchar(30) NOT NULL,
         PDiagnosis varchar(50) NOT NULL,
         HAddress varchar(60) NOT NULL,
         Hcity VARCHAR(20) NOT NULL
         );

CREATE
TABLE
Doctor(
        Doc-id varchar(50) PRIMARY KEY,
        Dname varchar(30) NOT NULL,
        Salary number(50) NOT NULL,
        Qualification varchar(60) NOT NULL,
        FOREIGN KEY (HospID) REFERENCES Hospital(HospID)
        );


CREATE TABLE
MedicalRecords(
```

Record-id varchar(50) PRIMARY KEY,
Date of examination Dt Date Format 'dd.mm,yyyy',
Qualification varchar(60) NOT NULL,
FOREIGN KEY (HospID) REFERENCES Patient(HospID)
);

CREATE TABLE
DoctorPatientattends(
FOREIGN KEY (Pat_ID) REFERENCES Patient(Pat_ID)
FOREIGN KEY (HospID) REFERENCES Patient(HospID)
PRIMARY KEY (Pat_Id, HospID)
);

[3 Marks]

6 b)

Description	Issue	Input	Output	Challenge
a. Conceptual Design Create model that captures major entities, relationships among entities, and attributes of entities required for a particular system.	- Capturing all data - Capturing relationships - Data integrity	- Functional specs - General understanding of problem	- ER diagram	
b. Logical Design Transform the major entity/attribute /relationship requirements into high level specification for database	- Providing location for all data - Data integrity	- ER diagram	- Relational database schema	-
b2 Improving Logical Design Improve the high-level database specification.	- Minimizing redundancy - Minimizing ambiguity	- Relational database schema	- Relational database schema	-
c. Physical Design Transform the high-level specifications for database into detailed specifications for how to construct actual database in a specific relational database software.	- Performance - Data integrity	- Relational database schema - Meaning of data	- Technical specifications for construction of the database	-

[10 Marks]

7 a)

i) $\sigma_{destination = \text{"New Delhi"}}$ (flight) **[1 Mark]**

ii) $\Pi_{fid}(\sigma_{pid = 123}(\text{booking}) \bowtie \sigma_{dest = \text{"Chennai"} \wedge fdate < 06/11/2020}(\text{flight}))$ **[1 Mark]**

iii) $\sigma_{src = \text{"Chennai"} \wedge dest = \text{"New Delhi"}}$ (flight) **[1 Mark]**

iv) Π_{pname} (passenger $\bowtie$ booking) **[1 Mark]**

v) $\Pi_{pname}((\Pi_{pid}(\text{passenger}) - \Pi_{pid}(\text{booking})) \bowtie \text{passenger})$ **[1 Mark]**

vi) $\Pi_{aname}(\text{agency} \bowtie_{acity = pcity}(\sigma_{pid = 123}(\text{passenger})))$ **[1 Mark]**

vii) $(\sigma_{fdate = 01/12/2020 \wedge time = 16:00}(flight)) \cap (\sigma_{fdate = 02/12/2020 \wedge time = 16:00}(flight))$ **[1 Mark]**

viii) $(\sigma_{fdate = 01/12/2020 \wedge time = 16:00}(flight)) \cup (\sigma_{fdate = 02/12/2020 \wedge time = 16:00}(flight))$ **[1 Mark]**

ix) $(\sigma_{fdate = 01/12/2020 \ AND \ time = 16:00}(flight)) \cup (\sigma_{fdate = 02/12/2020 \ AND \ time = 16:00}(flight))$ **[1 Mark]**

x) $\Pi_{aname}(agency \bowtie (\Pi_{aid}(agency) - \Pi_{aid}(\sigma_{pid = 123}(booking)))$ **[1 Mark]**

7 b a)

Triggers are database object. Basically, these are a special type of stored procedure that is automatically fired/executed when a DDL or DML command statement related to the trigger is executed. Triggers are used to assess/evaluate data before or after data modification using DDL and DML statements. These are also used to preserve data integrity, to control server operations, to audit a server and to implement business logic or business rule.

DDL Triggers

In SQL Server we can create triggers on DDL statements (like CREATE, ALTER, and DROP) and certain system-defined stored procedures that perform DDL-like operations.
Example: If you are going to execute the CREATE LOGIN statement or the sp_addlogin stored procedure to create login user, then both these can execute/fire a DDL trigger that you can create on CREATE_LOGIN event of SQL Server.
DDL trigger can be used to observe and control actions performed on the server, and to audit these operations. DDL triggers can be used to manage administrative tasks such as auditing and regulating database operations. **[2.5 Marks]**

DML Triggers

In SQL Server we can create triggers on DML statements (like INSERT, UPDATE, and DELETE) and stored procedures that perform DML-like operations. DML Triggers are of two types

1. **After Trigger (using FOR/AFTER CLAUSE)**

   This type of trigger fires after SQL Server finishes the execution of the action successfully that fired it.

Example: If you insert record/row in a table then the trigger related/associated with the insert event on this table will fire only after the row passes all the constraints, like as primary key constraint, and some rules. If the record/row insertion fails, SQL Server will not fire the After Trigger.

Instead of Trigger (using INSTEAD OF CLAUSE)

This type of trigger fires before SQL Server starts the execution of the action that fired it. This differs from the AFTER trigger, which fires after the action that caused it to fire. We can have an INSTEAD OF insert/update/delete trigger on a table that successfully executed but does not include the actual insert/update/delete to the table.
Example: If you insert record/row in a table then the trigger related/associated with the insert event on this table will fire before the row passes all the constraints, such as primary key constraint and some rules. If the record/row insertion fails, SQL Server will fire the Instead of Trigger.

```
CREATE TRIGGER trigger_name
ON {table|view}
```

```
[WITH ENCRYPTION|EXECUTE AS]
{FOR|AFTER|INSTEAD OF} {[CREATE|ALTER|DROP|INSERT|UPDATE|DELETE ]}
[NOT FOR REPLICATION]
AS
sql_statement [1...n ]
```

[2.5 Marks]

7 b b)

i) For each relation, the primary key must not contain any nulls. Room is related to Hotel through the attribute hotelNo. Therefore, the hotelNo in Room should either be null or contain the number of an existing hotel in the Hotel relation. In this problem, it would probably be unacceptable to have a hotelNo in Room with a null value.

[2 Marks]

ii) Booking is related to Hotel through the attribute hotelNo. Therefore, the hotelNo in Booking should either be null or contain the number of an existing hotel in the Hotel relation. However, because hotelNo is also part of the primary key, a null value for this attribute would be unacceptable. Similarly for guestNo. Booking is also related to Room through the attribute. **[3 Marks]**

8a)

1. ALTER TABLE DEPT RENAME TO DEPARTMENT; **[2 Marks]**

2. ALTER TABLE DEPARTMENT ADD(PINCODE NUMBER(6) NOT NULL);

DESC DEPARTMENT; **[2 Marks]**

3. ALTER TABLE DEPARTMENT DROP column LOC CASCADE CONSTRAINTS; **[2 Marks]**

4. ALTER TABLE DEPT RENAME COLUMN DNAME TO DEPT_NAME ; **[2 Marks]**

5. ALTER TABLE DEPARTMENT MODIFY LOC CHAR(10) ; **[2 Marks]**

8 b)

1. σmyear>1997(Movies) **[2 Marks]**

2. σmyear>1997 ∧ director='Hanson_(Movies) **[2 Marks]**

3. πtitle, rating(Movies) **[2 Marks]**

4. πactor(Actors) ∪ πdirector(Directors) **[2 Marks]**

5. e1 = πtitle(σactor='McDormand_ (Acts)) e2 = πtitle(σdirector='Coen_ (Movies)) result = e1 ∩ e2
[2 Marks]

9a)

If a flat-file database is poorly planned, denormalised and inconsistent, it will create problems when trying to insert, delete or modify the records (tables) in the database. This causes **anomalies**, which make handling the data increasingly difficult as the database grows. It also makes the data integrity harder to maintain. Trying to make the data consistent once an anomaly occurs can become quite difficult.

There are three types of problems that can occur in databases:

- **Insertion anomaly:** The database has been created in such a way that required data cannot be added unless another piece of unavailable data is also added. For example, a hospital database that cannot store the details of a new member until that member has been seen by a doctor.
- **Deletion anomaly:** The legitimate deletion of a record of data can cause the deletion of some required data. For example, deleting some of the patient's details can remove all the details of the patient from the hospital database.
- **Modification anomaly:** Incorrect data may have to be changed, which could involve many records having to be changed, leading to the possibility of some changes being made incorrectly.

[5 Marks]

For example, look at the below schema that represents information related to a school:

AccountInfo

ACCOUNTNO	ACCOUNTNAME	ACCOUNTADDRESS	CHILD	BALANCE
335684	Agatha Jones	Cape Town	Keene	R451
335684	Bob Jones	Cape Town	Keene	R451
335684	Agatha Jones	Cape Town	Joshua	R451
335684	Bob Jones	Cape Town	Joshua	R451

Problems:

- **Data redundancy:** Data redundancy refers to storing the exact same data at different places in a database. Data redundancy increases the size of your database, creates integrity problems, decreases the efficiency of the database and can lead to database anomalies. Data should be stored in such a way that it should not be repeated in multiple tables. The family can be represented as one unit.

- **Update anomaly:** Data will be inconsistent if one entity is updated, for example, if the family address changes, the data capturer will need to change all four entities.
- **Unable to represent some information:** the school cannot keep information about learners who are on the waiting list as they do not have an account with the school.
- **Deletion anomaly:** closure of the account for one child will remove all data of the second child from the database. **[2 Marks]**

To prevent anomalies you need to normalise the database by efficiently organising the data in a database.
According to Edgar F Codd, the inventor of relational databases, the goals of normalisation include:

- removing all redundant (or repeated) data from the database
- removing undesirable insertions, updates and deletion dependencies
- reducing the need to restructure the entire database every time new fields are added to it
- making the relationships between tables more useful and understandable.

Normalisation is a systematic approach of decomposing tables to eliminate data redundancy and Insertion, Modification and Deletion Anomalies. The database designer structures the data in a way that eliminates unnecessary duplication(s) and provides a rapid search path to all necessary information. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables. This process of specifying and defining tables, keys, columns, and relationships in order to create an efficient database is called **normalisation**. **[3 Marks]**

9b) a) From the formal Definition of Functional Dependency : Functional dependency in DBMS refers to a relationship that is present between attributes of any table that are dependent on each other.

For the figure below : -

A	B	C
1	2	3
2	2	2
1	3	2
4	2	3

Does A → B holds , we need to check whether for each unique value of A it should have same value for B in the relation.

Now from the above figure, For Value of 1 in A , B have two different values thus A → B does not hold as per the information given in above figure. [2 Marks]

Now for BC → A holds , we need to check whether for each unique value of BC it should have same value for A in the relation.

Now from the above figure, For Value of 2 3 in BC , A have two different values thus BC → A does not hold as per the information given in above figure. [2.Marks]

9b b)

For given relation R(ABCDEH) with following FDs , find all the relevant candidate keys

{A→ B, BC→ D, E→ C,D→ A}

Applying the closure set of property and finding the closure of each attribute.

An attribute or set of attribute whose closure finds all the elements will be the candidate key for the given relation

A^+ = [A,B]

B^+ **=** [B]

C^+ **=** [C]

D^+ **=** [D,A,B]

E+ = [E,C]

H^+ **=** [H]

As we can see that none of single attribute closure is not sufficient to find all the elements on right hand side. So now let us try to find the closure with combination of Two attributes

AB^+ = [A,B]

AC^+ = [A,B,D]

AD^+ = [A,B,D]

AE^+ = [A,B,C,D]

AH^+ = [A,B,H]

Similarly for B combinations

BC^+ = [B,C,D,A]

BD^+ = [B,D,A]

BE^+ = [B,E,C,D]

BH^+ = [B,H]

Similarly for C combinations

CD^+ = [C,D,A,B]

CE^+ = [E,C]

CH^+ = [C,H]

Similarly for D combinations

DE^+ = [E,D,A,B,C]

DH^+ = [D,H,A,B,C]

Similarly for E combinations

EH^+ = [E,H,C] **[3 Marks]**

Still we are not able to find the set of all attributes with combinations of two attributes closure, now apply the same procedure for three

$[ABC]^+$ = [A.B.C.D]

$[ABD]^+$ = [A B C D]

$[ABE]^+$= [A B E C D]

$[ABH]^+$ =[A,B,H]

$[ACD]^+$ =[A,B,C,D,H]

$[ACE]^+$ =[A,B,C,D]

$[ACH]^+$ =[A,B,C,D,H]

$[ADE]^+$ =[A,B,C,D,E]

$[ADH]^+$ =[A,B,H]

$[AEH]^+$ =[A,B,C,D,E,H] **As we are able to find all the attribute, one of the Candidate key is AEH**

Applying the same formula, other Candidate key are

BEH, DEH

So Total 3 Candidate key are possible for above relations (AEH, BEH, DEH) **[3 Marks]**

ABES Engineering College, Ghaziabad

B.Tech Odd Semester Sessional Test-1

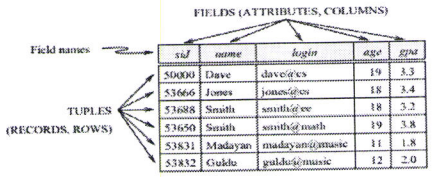
Printed Pages: 3
Session:2022-2023
Roll No.
Date of Exam:17.10.22
Time:

Course Code: KCS 501

Course Name: Database Management System

Maximum Marks: 75

Instructions:
1. Attempt All sections.
2. If require any missing data, then choose suitably.

Q.No.	Question	Marks	CO	KL	PI
	Section-A				
1	**Attempt ALL Parts**	(5x2=10)			
a)	Consider a relational table with a single record for each registered student with the following attributes. 1. Registration_Number: Unique registration number of each registered student 2. UID: Unique Identity number, unique at the national level for each citizen 3. BankAccount_Number: Unique account number at the bank. A student can have multiple accounts or joint accounts. This attributes stores the primary account number 4. Name: Name of the Student 5. Hostel_Room: Room number of the hostel **[GATE 2020]** Find the candidate key	2	CO1	K2	1.4.1
b)	Let E1 and E2 be two entities in an ER diagram with simple single-valued attributes. R1 and R2 are two relationships between E1 and E2, where R1 is one-to-many and R2 is many-to-many. R1 and R2 do not have any attributes of their own. What is the minimum number of tables required to represent this situation in the relational model? **[GATE 2005]**	2	CO1	K2	2.1.3
c)	Consider the instance of the Students relation shown in Figure below  sid name login age gpa 50000 Dave dave@cs 19 3.3 53666 Jones jones@cs 18 3.4 53688 Smith smith@ee 18 3.2 53650 Smith smith@math 19 3.8 53831 Madayan madayan@music 11 1.8 53832 Guldu guldu@music 12 2.0 Give an example of an attribute (or set of attributes) that you can deduce is not a candidate key, based on this instance being legal.	2	CO2	K3	2.3.1
d)	Assume that, in addition to storing the grade a student has received in every course he has completed, the system stores the student's cumulative GPA. Describe an integrity constraint that relates this information. Describe how the constraint would be violated if the transaction that records a new grade were not atomic.	2	CO2	K3	2.3.1
e)	In a schema with attributes A, B, C, D and E following set of functional dependencies are given $A \rightarrow B$ $A \rightarrow C$ $CD \rightarrow E$ $B \rightarrow D$ $E \rightarrow A$. Find the set(s) of Functional Dependency which is implied from the above set **[GATE 2005]**	2	CO3	K3	2.3.1
	Section-B				
2	**Attempt ANY ONE part from the following**	(1x5=5)			
a)	Illustrate the type of database users	5	CO1	K1	1.3.1
b)	Sketch difference among Super key, Candidate key,Primary key, Foreign key and surrogate key.	5	CO1	K2	1.3.1
3	**Attempt ANY ONE part from the following**	(1x5=5)			
a)	Analyse, briefly, the features of different data models.	5	CO2	K1	1.4.1
b)	Demonstrate with a neat diagram, the structure of a DBMS.	5	CO2	K1	1.4.1
4	**Attempt ANY ONE part from the following**	(1x5=5)			
a)	Given a relational schema R(X, Y, Z, W, V) set of functional dependencies P and Q such that: P = { $X \rightarrow Y$, $XY \rightarrow Z$, $W \rightarrow XZ$, $W \rightarrow V$} and Q = { $X \rightarrow YZ$, $W \rightarrow XV$ } using FD sets P and Q analyse the equivalence of two sets?	5	CO3	K3	2.3.1
b)	Consider the relation shown in Figure below  X Y Z x_1 y_1 z_1 x_1 y_1 z_2 x_2 y_1 z_1 x_2 y_1 z_3 1) List all the functional dependencies that this relation instance satisfies **[2.5]** 2) Assume that the value of attribute Z of the last record in the relation is changed from z3 to z2. Now list all the functional dependencies that this relation instance satisfies **[2.5]**	2.5+ 2.5	CO3	K2	2.3.1

Section-C					
5	**Attempt ANY ONE part from the following**	**(1x10=10)**			
a)	A university registrar's office maintains data about the following entities: i) Courses, including number, title, credits, syllabus, and prerequisites; ii) Course offerings, including course number, year, semester, section number, instructor(s), timings, and classroom; iii) Students, including student-id, name, and program; and iv) Instructors, including identification number, name, department, and title. Further, the enrollment of students in courses and grades awarded to students in each course they are enrolled for must be appropriately modeled. Construct an E-R diagram for the registrar's office. Document all assumptions that you make about the mapping constraints.	**2.5X4**	**CO1**	**K3**	**2.1.3**
b)	a) Illustrate difference between traditional file processing systems and database systems. **[5]** b) Examine, briefly, the importance of different level of data independence with suitable example. **[5]**	**5+5**	**CO1**	**K2**	**1.4.1**
6	**Attempt ANY ONE part from the following**	**(1x10=10)**			
a)	Sketch an ER diagram of Hospital or Bank with showing the Specialization, Aggregation, Generalization. Also convert it in to relational schemas and SQL DDL.	**10**	**CO1**	**K2**	**1.4.1**
b)	Imagine that you have been assigned to a team that will be developing an inventory tracking system. As part of the project startup, your manager has asked each team leader to bring a basic work plan to the next meeting. At that meeting, these work plans will be analyzed to determine the overall project timeframe, costs, personnel requirements, and software requirements. For now, as the team leader for the data design team, you have been asked to bring a work plan that identifies the phases of data design and includes the following information for each phase: a). a description of the data design phase, b). the inputs of the phase, c). the outputs of the phase, d). a key issue addressed in the phase e). a challenge that you can anticipate would occur in the phase. Please prepare the response you will bring to the meeting.	**10**	**CO1**	**K3**	**2.3.1**
7	**Attempt ANY ONE part from the following**	**(1x10=10)**			
a)	Consider the following relational database schema consisting of the four relation schemas: passenger (pid, pname, pgender, pcity) agency (aid, aname, acity) flight (fid, fdate, time, src, dest) booking (pid, aid, fid, fdate) Answer the following questions using relational algebra queries; i) Get the complete details of all flights to New Delhi. ii) Find only the flight numbers for passenger with pid 123 for flights to Chennai before 06/11/2020. iii) Get the details about all flights from Chennai to New Delhi. iv)Find the passenger names for passengers who have bookings on at least one flight. v)Find the passenger names for those who do not have any bookings in any flights. vi)Find the agency names for agencies that located in the same city as passenger with passenger id 123. vii)Get the details of flights that are scheduled on both dates 01/12/2020 and 02/12/2020 at 16:00 hours. viii) Get the details of flights that are scheduled on either of the dates 01/12/2020 or 02/12/2020 or both at 16:00 hours. ix) Get the details of flights that are scheduled on either of the dates 01/12/2020 and 02/12/2020 and both at 16:00 hours. x) Find the agency names for agencies who do not have any bookings for passenger with id 123.	**1X10**	**CO2**	**K3**	**2.1.3**
b)	a) Ellobrate the siginficance of different types of trigger with proper example **[5]** b) Consider the following database schema. **Hotel** (hno, hname, hcity) **Room** (rno, hno, type, price) **Booking** (hno, gno, datefrom, dateto, rno) **Guest** (gno, guestname, guestadd) Answer the following: a. Identify the foreign keys in the schema. b. Explain how referential integrity rules can be applied to these relations. **[5]**	**5+5**	**CO2**	**K2**	**2.1.3**

8	Attempt ANY ONE part from the following	(1x10=10)			
a)	Consider Dept table DEPTNO DNAME LOC Perform the following: i) Rename the table dept as department ii) Add a new column PINCODE with not null constraints to the existing table DEPT iii) Rename the column DNAME to DEPT_NAME in dept table iv) Change the data type of column loc as CHAR with size 10 v). Delete table	2X5	CO2	K2	1.4.1
b)	Consider the MOVIE DATABASE Write following relational algebra queries for a given set of relations. 1. Find movies made after 1997 2. Find movies made by Hanson after 1997 3. Find all movies and their ratings 4. Find all actors and directors 5. Find Coen's movies with McDormand Movies title | director | myear | rating Fargo | Coen | 1996 | 8.2 Raising Arizona | Coen | 1987 | 7.6 Spiderman | Raimi | 2002 | 7.4 Wonder Boys | Hanson | 2000 | 7.6 Actors actor | ayear Cage | 1964 Hanks | 1956 Maguire | 1975 McDormand | 1957 Acts actor | title Cage | Raising Arizona Maguire | Spiderman Maguire | Wonder Boys McDormand | Fargo McDormand | Raising Arizona McDormand | Wonder Boys Directors director | dyear Coen | 1954 Hanson | 1945 Raimi | 1959	2X5	CO2	K2	1.4.1

9	Attempt ANY ONE part from the following	(1x10=10)			
a)	Illustrate out the different type of anomailes assocated with database design? Provide the mechanism how to remove such anomlaies with proper example	10	CO3	K2	1.4.1
b)	a) The following is an instance of a relation schema. Can you tell whether the schema includes the functional dependencies A → B and BC → A? **[4]** A | B | C 1 | 2 | 3 2 | 2 | 2 1 | 3 | 2 4 | 2 | 3 b) For given relation R(ABCDEH) with following FDs , find all the relevant candidate keys {A→ B, BC→ D, E→ C,D→ A} **[6]** **GATE 2005**	4+6	CO3	K2	1.4.1

CO Course Outcomes mapped with respective question
KL Bloom's knowledeg Level (K1, K2, K3, K4, K5, K6)
K1- Remember, K2- Understand, K3-Apply, K4- Analyze, K5: Evaluate, K6- Create