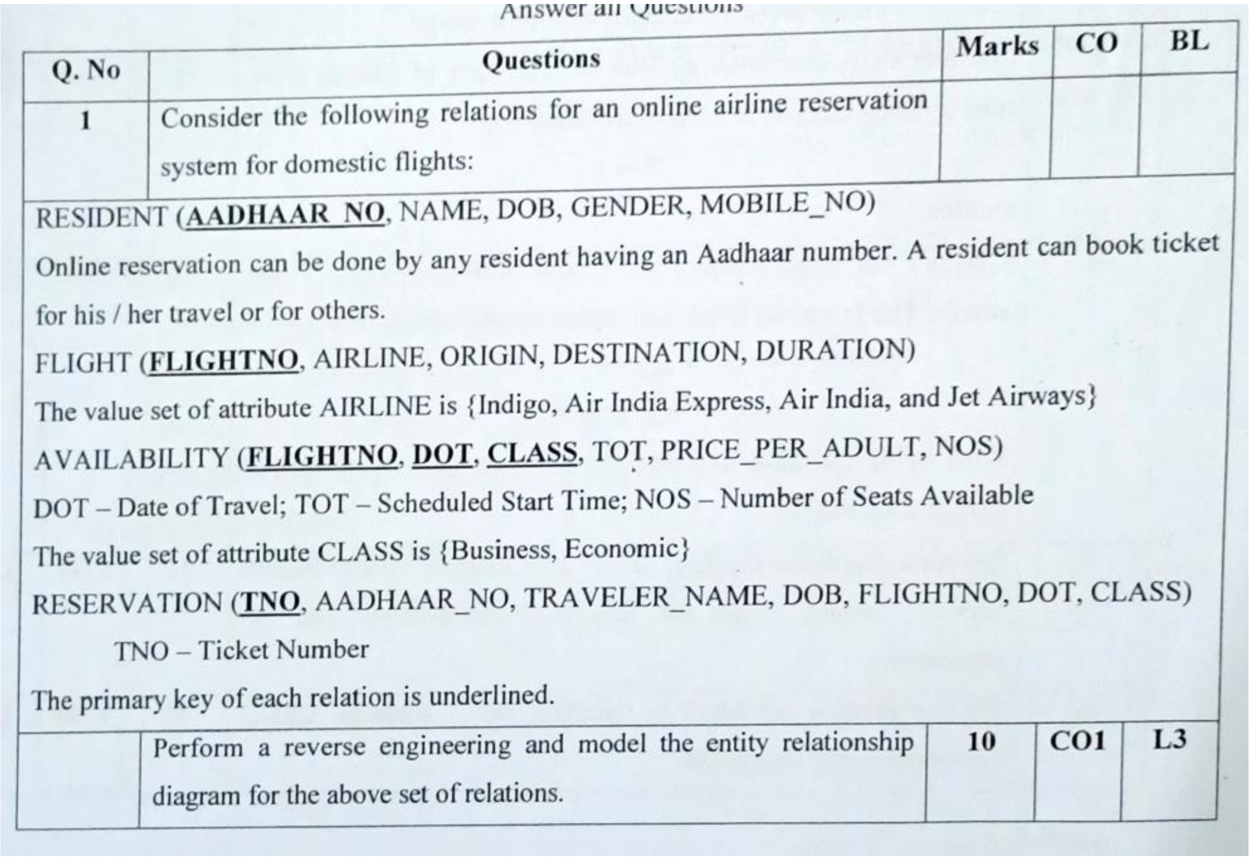
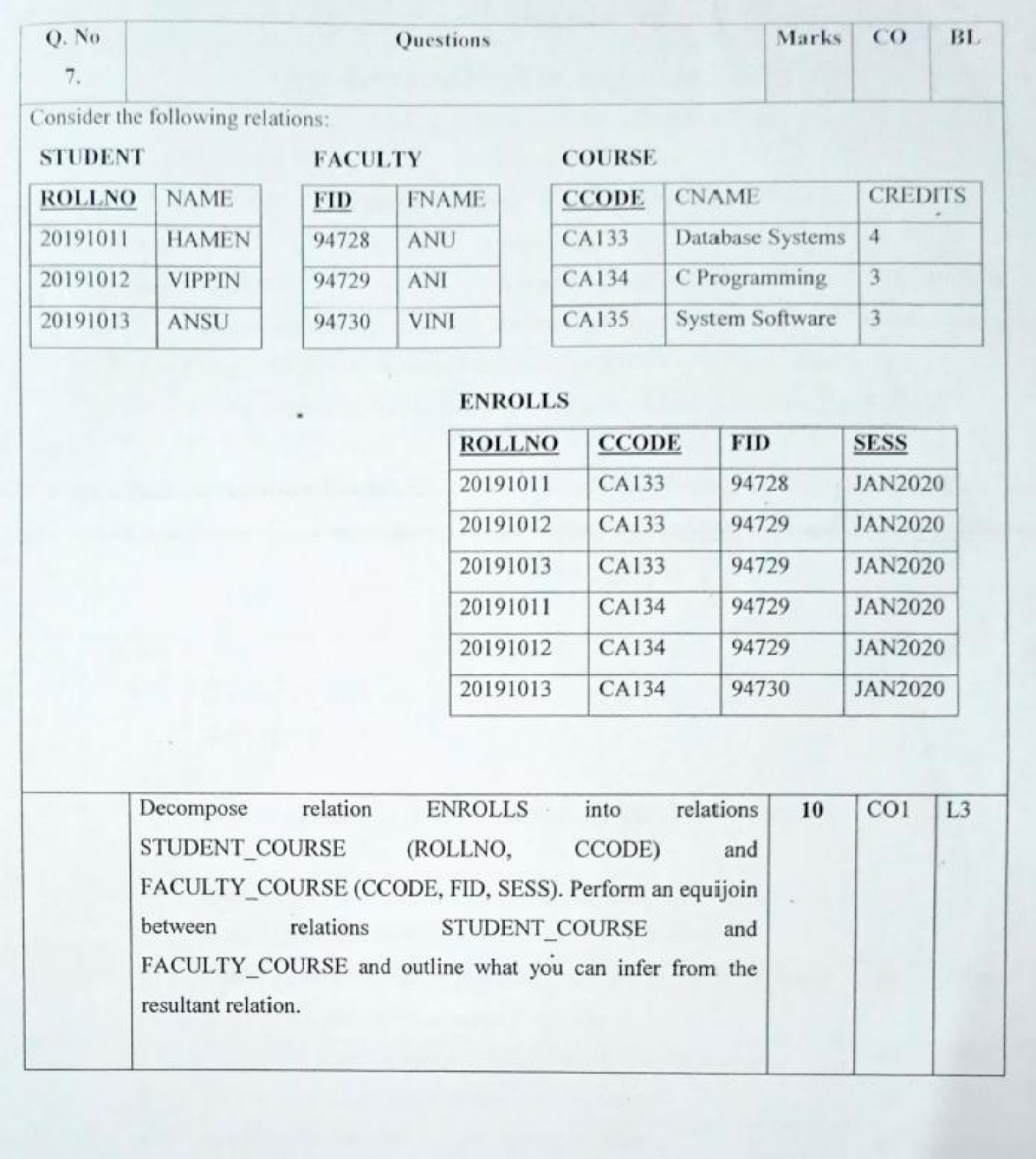


Answer all Questions

<table>
<tr><th>Q. No</th><th>Questions</th><th>Marks</th><th>CO</th><th>BL</th></tr>
<tr><td>1</td><td>Consider the following relations for an online airline reservation system for domestic flights:</td><td></td><td></td><td></td></tr>
<tr><td colspan="5">RESIDENT (<b>AADHAAR_NO</b>, NAME, DOB, GENDER, MOBILE_NO)
Online reservation can be done by any resident having an Aadhaar number. A resident can book ticket for his / her travel or for others.
FLIGHT (<b>FLIGHTNO</b>, AIRLINE, ORIGIN, DESTINATION, DURATION)
The value set of attribute AIRLINE is {Indigo, Air India Express, Air India, and Jet Airways}
AVAILABILITY (<b>FLIGHTNO</b>, <b>DOT</b>, <b>CLASS</b>, TOT, PRICE_PER_ADULT, NOS)
DOT – Date of Travel; TOT – Scheduled Start Time; NOS – Number of Seats Available
The value set of attribute CLASS is {Business, Economic}
RESERVATION (<b>TNO</b>, AADHAAR_NO, TRAVELER_NAME, DOB, FLIGHTNO, DOT, CLASS)
TNO – Ticket Number
The primary key of each relation is underlined.</td></tr>
<tr><td></td><td>Perform a reverse engineering and model the entity relationship diagram for the above set of relations.</td><td>10</td><td>CO1</td><td>L3</td></tr>
</table>

2.	Consider three entities STUDENT, BRANCH and DEPARTMENT. The attributes of STUDENT entity are ROLLNO and NAME. The attributes of BRANCH entity are BCODE and BNAME. The attributes of DEPARTMENT entity are DCODE and DNAME. The STUDENT entity and the DEPARTMENT Entity are associated with a N:1 relationship BELONGS TO. The STUDENT entity and the BRANCH entity are associated with a N:1 relationship ADMITTED TO. The DEPARTMENT entity and the BRANCH entity are associated with a 1: N relationship HAS.			
	i. Model an entity relationship diagram and map the entity relationship diagram to relations.	5	CO1	L3
	ii. Check whether each relation is normalized, if yes justify. If a relation is not normalized normalize the relation.	5	CO1	L3
	iii. If a relation is not normalized state reasons from the viewpoint of the entity relationship model.	5	CO1	L3
3.	Consider the relation $\alpha\ (\beta,\gamma)$. The attributes β,γ of relation α are atomic. What can you infer from the following? $\beta \twoheadrightarrow \gamma$ Outline.	2	CO1	L3
4.	Consider the relation $\mu(\alpha,\beta,\gamma,\delta)$. The attributes $\alpha,\beta,\gamma,\delta$ are atomic. The following functional dependencies hold: $\alpha,\beta \rightarrow \gamma$ $\alpha \rightarrow \delta$ What is the candidate key of relation μ? Is relation μ in second normal form? Outline.	3	CO1	L3
5.	Present a distributed database design for a chain of "Maruti Suzuki Service Centers". State the functional requirements you are considering.	10	CO2	L3
6.	For the relation presented in Question No. 1 illustrate CRUD operations using MongoDB.	10	CO3	L3



Q. No 7.	Questions	Marks	CO	BL

Consider the following relations:

STUDENT

ROLLNO	NAME
20191011	HAMEN
20191012	VIPPIN
20191013	ANSU

FACULTY

FID	FNAME
94728	ANU
94729	ANI
94730	VINI

COURSE

CCODE	CNAME	CREDITS
CA133	Database Systems	4
CA134	C Programming	3
CA135	System Software	3

ENROLLS

ROLLNO	CCODE	FID	SESS
20191011	CA133	94728	JAN2020
20191012	CA133	94729	JAN2020
20191013	CA133	94729	JAN2020
20191011	CA134	94729	JAN2020
20191012	CA134	94729	JAN2020
20191013	CA134	94730	JAN2020

	Decompose relation ENROLLS into relations STUDENT_COURSE (ROLLNO, CCODE) and FACULTY_COURSE (CCODE, FID, SESS). Perform an equijoin between relations STUDENT_COURSE and FACULTY_COURSE and outline what you can infer from the resultant relation.	10	CO1	L3

Q1: Reverse Engineering and ER Diagram for Airline Reservation System

Entities and Attributes

1. **RESIDENT** (Aadhaar_No (PK), Name, DOB, Gender, Mobile_No)
2. **FLIGHT** (FlightNo (PK), Airline, Origin, Destination, Duration)
3. **AVAILABILITY** (FlightNo (FK), DOT, Class, TOT, Price_Per_Adult, NOS, **PK: (FlightNo, DOT, Class)**)
4. **RESERVATION** (TNO (PK), Aadhaar_No (FK), Traveler_Name, DOB, FlightNo (FK), DOT, Class)

Relationships

- **A Resident can make multiple Reservations.** (1:N)
- **A Flight has multiple availabilities for different dates and classes.** (1:N)
- **A Reservation is for a specific Flight, Date, and Class.** (N:1)

ER Diagram

Draw an ER diagram with entities connected by the relationships as described above.

Q2: Entity Relationship for STUDENT, BRANCH, DEPARTMENT

(i) ER Model Mapping

- **STUDENT** (RollNo (PK), Name)
- **BRANCH** (BCode (PK), BName)
- **DEPARTMENT** (DCode (PK), DName)
- **RELATIONSHIPS**:
  - **BELONGS_TO** (Student → Department) (N:1)
  - **ADMITTED_TO** (Student → Branch) (N:1)
  - **HAS** (Department → Branch) (1:N)

(ii) Normalization

- If any redundancy exists, convert it into normalized relations.

(iii) Justification

- If the relations are in **1NF** (Atomicity), **2NF** (No Partial Dependency), **3NF** (No Transitive Dependency), then they are normalized.

Q3: Functional Dependency and Normalization

(i) Given Relation α(β, γ)

- **$\beta \to \gamma$** means β functionally determines γ.
- If β is a candidate key, the relation is already in **BCNF**.

(ii) Given Relation μ(α, β, γ, δ)

- Functional Dependencies:
  - **$\alpha, \beta \to \gamma$**
  - **$\alpha \to \delta$**
- Candidate Key: **(α, β)**
- **Check 2NF**:
  - If partial dependency exists (like $\alpha \to \delta$), it is not in **2NF**.
- **Normalization to 2NF**:
  - Decompose into **μ1(α, δ)** and **μ2(α, β, γ)**.

Q4: Distributed Database Design for Maruti Suzuki Service Centers

Functional Requirements

1. **Branch-wise service centers** with local data storage.
2. **Centralized database** for inventory and service tracking.
3. **Distributed transaction management** for customer bookings.
4. **Replication and consistency management** for multiple branches.

Q5: CRUD Operations in MongoDB

For the given database schema (from Q1):

1. **Create:**

```
db.RESERVATION.insertOne({
    TNO: "T12345",
    Aadhaar_No: "123456789012",
    Traveler_Name: "John Doe",
    DOB: "1995-05-20",
    FlightNo: "AI101",
    DOT: "2025-04-01",
    Class: "Economy"
});
```

2. **Read:**

```
db.RESERVATION.find({ FlightNo: "AI101" });
```

3. **Update:**

```
db.RESERVATION.updateOne(
    { TNO: "T12345" },
    { $set: { Class: "Business" } }
);
```

4. **Delete:**

```
db.RESERVATION.deleteOne({ TNO: "T12345" });
```

Q7: Decomposing ENROLLS into Two Relations and EquiJoin

Decomposition

1. **STUDENT_COURSE (ROLLNO, CCODE)**

```
(20191011, CA133)
(20191012, CA133)
(20191013, CA133)
(20191011, CA134)
(20191012, CA134)
(20191013, CA134)
```

2. **FACULTY_COURSE (CCODE, FID, SESS)**

```
(CA133, 94728, JAN2020)
(CA133, 94729, JAN2020)
(CA134, 94729, JAN2020)
(CA134, 94729, JAN2020)
(CA134, 94730, JAN2020)
```

Performing EquiJoin on CCODE

- The result will restore the original ENROLLS table.