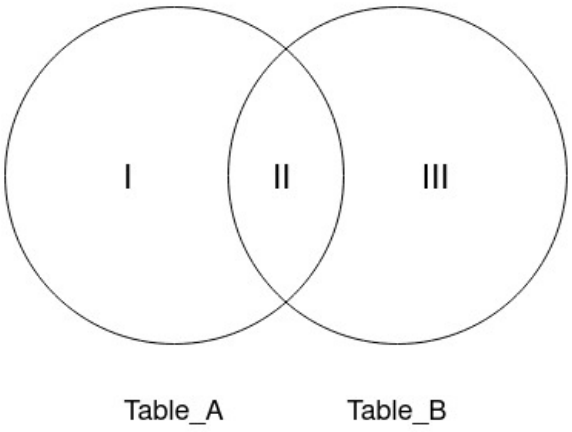
Section 1

Consider the following venn diagram for two tables which can be considered as a visual representation of the result of join of the tables:



For each of the following SQL statements, mention all the regions (among I, II and III) that will be included as a part of the result of the statement and **justify** your answer (Answers without justification will be given 0 marks): (1 + 2 + 2)

1. `SELECT <select_list> FROM Table_A A INNER JOIN Table_B B ON A.Key = B.Key;`

   Region: II

2. `SELECT <select_list> FROM Table_A A FULL OUTER JOIN Table_B B ON A.Key = B.Key WHERE A.Key IS NULL OR B.Key IS NULL;`

   Region: I, III

3. `SELECT <select_list> FROM Table_A A RIGHT JOIN Table_B B ON A.Key = B.Key WHERE A.Key IS NULL;`

   Region: III

Section 2

Consider the following scenarios. For each of the scenarios, mention if any informal database design principle is being violated. If being violated, suggest a way to remedy the violation. (3 + 3 + 3)

1. The following schema is used in a database to store student details:

   STUDENT (StudentID, Name, DOB, CGPA, Major, AdvisorID)

   However, it has been observed that only 10% of the students actually have advisors. All the other students are single degree students who don't have any advisors.

   Lot of NULL values in the table, We can make an additional table R2(Student_ID, AdvisorID) which will contain only those students who have advisor and original table can be R(StudentID, Name, DOB, CGPA, Major)

2. The following schema is used to store the details of movie in a database:

   Movie (MovieID, Title, DirectorName, DirectorContact, Year, Genre, Rating)

   The tuple where MovieID was '1MN2F' was deleted because of some legal conflict. It happens to be the only tuple which consisted of the Director 'Brad Davis'. Now, the information about 'Brad Davis' is also lost.

   Deletion Anomaly, Should store director details in separate table

3. Consider the following schema of transaction table:

   TRANSACTION (OrderID, TransactionID, Amount, Discount, TransactionDate, PaymentMethod)

   It is decomposed into two tables as follows:

   TRANSACTION_1 (OrderID, TransactionID, TransactionDate)
   TRANSACTION_2 (TransactionDate, Amount, Discount, PaymentMethod)

Whenever information about the payment method of a transaction is needed, a JOIN is performed between the two tables TRANSACTION_1 & TRANSACTION_2.

Spurious tuple generation
Should use Transaction ID in second table as joining attribute

Section 3

Consider the following database schema for a flight management system.

- **Flight** (`fid: varchar, origin_airport_id: int, destination_airport_id: int, departure: datetime, arrival: datetime, capacity: int, price: float`)
  - Primary key: `fid`
  - Foreign Keys: `origin_airport_id` and `destination_airport_id` reference Airport's `aid`
- **Airport** (`aid: int, name: varchar, city: varchar, country: varchar`)
  - Primary key: `aid`
- **Passenger** (`pid: int, name: varchar, passport: varchar, dob: date, gender: char`)
  - Primary key: `pid`
- **Ticket** (`tid: int, passenger_id: int, flight_id: varchar, seat: varchar, status: varchar, date: date`)
  - Primary key: `tid`
  - Foreign keys: `passenger_id` references Passenger's `pid`, `flight_id` references Flight's `fid`
- **Crew** (`cid: int, name: varchar, role: varchar, salary: float, flight_id: varchar`)
  - Primary key: `cid`
  - Foreign key: `flight_id` references Flight's `fid`

1. Write a SQL query to find the name and role of the crew members who are working on the flight with the highest capacity. (2)

```
SELECT C.name, C.role FROM (Crew C INNER JOIN Flight F ON C.flight_id
= F.fid) WHERE F.capacity = (SELECT MAX(capacity) FROM Flight);
```

2. Write a SQL query to find the total number of tickets sold for each flight (Display flight id and total number of tickets sold for that flight). (2)

```
SELECT F.fid, COUNT(*) AS tickets_sold FROM (Flight F LEFT JOIN
Ticket T ON F.fid = T.flight_id) GROUP BY F.fid;
```

3. Write a SQL query to find the id and departure time of the flights that depart later than any flight from the city 'Hyderabad'. (3)

```
SELECT F.fid, F.departure FROM Flight F WHERE F.departure > (SELECT
MAX(departure) FROM (Flight INNER JOIN Airport ON
origin_airport_id=aid) WHERE city='Hyderabad');
```

OR (depending upon the interpretation of 'any')

```
SELECT F.fid, F.departure FROM Flight F WHERE F.departure > (SELECT
MIN(departure) FROM (Flight INNER JOIN Airport ON
origin_airport_id=aid) WHERE city='Hyderabad');
```

4. Write a SQL query to find the count of each gender who have booked a ticket for a flight that has has an id starting with 'IN'. (3)

```
SELECT P.gender, COUNT(P.pid) AS gender_count FROM ((Passenger P
INNER JOIN Ticket T ON P.pid = T.passenger_id) INNER JOIN Flight F ON
T.flight_id = F.fid) WHERE F.fid LIKE 'IN%' GROUP BY P.gender;
```

Section 4

1. Let R(A, B, C, D, E, F) be a relational scheme with the following functional dependencies:

- C → F
- E → A
- EC → D
- A → B

Prove or disprove: EC is a key for R. (2)

Proof:

EC is a key for R. Closure of EC = {ECF} (C → F) = {ECFA} (E → A) = {ECFAD} (EC → D) = {ECFADB} (A → B).

All the attributes are functionally dependent on EC. Hence, EC is a key for R.

2. Let R(A, B, C, D, E) be a relational scheme with the following functional dependencies:

- A → B
- A → C
- CD → E
- B → D
- E → A

State True or False for the following statements with explanation (Answers without any explanation with receive 0 marks): (2 + 2)

- CD → AC is implied from the above set of FDs.
- BD → CD is implied from the above set of FDs.

- CD → AC is true as CD → E and E → A
- BD → CD is not true as it cannot be derived from the given set of FDs.