Assignment 2: Relational Model

● Graded

Group

Nipun Sanjay Khivansara
Krishiv Shreyans Dakwala
View or edit group

Total Points

95 / 100 pts

Question 1

Part 1 — **30** / 30 pts

1.1 A — **10** / 10 pts

✔ **– 0 pts** Correct

– 2 pts Part of answer is incorrect

1.2 B — **10** / 10 pts

✔ **– 0 pts** Correct

– 2 pts Part of the answer is incorrect

– 4 pts Part of the answer is incorrect

1.3 C — **10** / 10 pts

✔ **– 0 pts** Correct

– 2 pts Part of answer is incorrect

– 4 pts Part of answer is incorrect

Question 2

Part 2 — **65** / 70 pts

– 5 pts Relation missing

– 3 pts Improper foreign key constraint

– 3 pts Attribute missing

– 3 pts foreign Key constraint missing

– 2 pts Unnecessary relation

– 0 pts Correct

– 5 pts Improper SQL order

✔ **– 5 pts** Incomplete primary keys

Primary keys for Location and Usage Readings are missing in SQL

Question assigned to the following page: 1.1

DBMS Assignment 2

Team:
Nipun Khivansara(57902188)
Krishiv Dakwala(94282115)

Part 1

Consider a relation schema R(A, B, C, D, E, G) and the following sets of functional dependencies
F1 = {ABC → D, BC → EA, BCE → G}
F2 = {A→B, C→AD, AE→CG, BC → C}
F3 = {AC → B, BC → D, BD → E, AE → G, ED → A, DA → C}

A. For each set F1, F2 and F3 :-
Work out whether relation R with the respective set is in BCNF and show how you reached the answer.

Solution:

1) **F1** : True
Candidate key -> {BC}

1NF - True
2NF - True (The table should not have partial dependency on the candidate key)
3nf - True (No transitive dependency as BC is at the left side of all equations}
BCNF - True (All equations satisfy the criteria super key -> non prime attributes**)**

---

2) **F2** : False
Candidate Key -> {CE, AE}

1NF -> True
2NF -> In relation C -> AD, D is dependent on a subset of the candidate key(CE) hence not in 2nf
BCNF -> Not in BCNF as it is not in 2NF and criteria super key -> non prime attributes not satisfied for relation A -> B

---

3) **F3**: False
Candidate Key -> {AC},{ED},{DA},{BC},{BD}

1NF -> True
2NF -> True
3NF -> True
BCNF -> False, Not in BCNF (AE->G) violates the condition as AE is not a candidate key

Question assigned to the following page: 1.2

B. For each set F1, F2 and F3 :-
Find all candidate keys of R and show how you reached the answer.

Solution:

1) **F1** ->

Given: F1 = {ABC ⟶ D, BC ⟶ EA, BCE ⟶ G}
Attributes that can be determined (Are on the right side of the relation): ADEG
Attributes that cannot be determined: BC
Hence we start by checking BC
{BC} -> {BCEA}, now we check BCEA
{BCEA} -> {BCEADG}
Hence BC is a candidate key.

Hence Candidate Key : {BC}

---

2) **F2** ->

Given: F2 = {A⟶B, C⟶AD, AE⟶CG, BC ⟶ C}
Attributes that can be determined (Are on the right side of the relation): ABCDG
Attributes that cannot be determined: E
Hence we start by checking E
{E} -> Cannot determine anything alone
We now check AE
{AE} -> {AECGB}
{AECG} -> { AECGBD} Hence AE is a candidate key

Now we check CE
{CE} -> {CEAD}
{CEAD} -> {CEADBG} Hence CE is also a candidate key

Hence Candidate Key : {AE,CE}

---

3) **F3** ->

Given: F3 = {AC ⟶ B, BC ⟶ D, BD ⟶ E, AE ⟶ G, ED ⟶ A, DA ⟶ C}

Since all the attributes can be determined (are on the right hand side of the relations) we check the candidate keys manually.

{AC} -> {ABCDG}
{ED} -> {ABCDG}
{DA} -> {ABCDG}

Question assigned to the following page: 1.3

{BC} -> {ABCDG}
{BD} -> {ABCDG}

Hence Candidate Key : {AC},{ED},{DA},{BC},{BD}

C. For each set F1, F2 and F3 :-
Consider partitioning R into 3 sub relations R1{A,B,C} , R2{D,E,G} , R3{B,C,D}. Is this decomposition lossless? Explain your answer.

1) Given: F1 = {ABC → D, BC → EA, BCE → G}

   Check if R1 ∩ R3. R1 ∩ R3 = {BC}.

   BC is a candidate key of both R1 and R3. Hence lossless decomposition

   Check R13 and R2. R13 ∩ R2 = {D}

   D is not the candidate key of R2 or R13. So the decomposition is lossy

   R1 ∩ R2. R1 ∩ R2 = {∅}. As it is a null set, the decomposition is lossy

   Checking R2, R3. R2 ∩ R3 = {D}

   As D is not a candidate key of either R3 or R2. We can say that the decomposition is lossy.

   Since all individual decompostions are lossy, we can say that the decomposition R1,R2,R3 is lossy.

---

2) Given: F2 = {A→B, C→AD, AE→CG, BC → C} Candidate key = {AE,CE}

   First, we check R1 ∩ R2. R1 ∩ R2 = {∅}. As the intersection of R1 and R2 is a null set, the decomposition is lossy.

   Now, we check R1 and R3. R1 ∩ R3 = {BC}.
   {BC}+ = {A, B, C, D}. We can see that BC is candidate key of both R1 and R3. Hence the decomposition R1 and R3 is lossless.

   Now, we check R13 = {A, B, C, D} and R2. R13 ∩ R2 = {D}
   {D}+ = {D}. Thus, we can see that D is not a candidate key of either R13 or R2. So, the decomposition R13 and R2 is lossy.

   Now, we check R2 ∩ R3. R2 ∩ R3 = {D}.
   {D}+ = {D}. Thus, we can see that D is not a candidate key of either R3 or R2. Hence the decomposition R2 and R3 is lossy.

Question assigned to the following page: 1.3

Since all individual decompositions are lossy, the decomposition R1, R2 and R3 is lossy.

---

3) Given: F3 = {AC → B, BC → D, BD → E, AE → G, ED → A, DA → C}
Firstly, check R1 ⋈ R3. R1 ∩ R3 = {BC}.
{BC}+ = {A, B, C, D, E, G}. BC is candidate key of both R1 and R3. So, the decomposition R1 and R3 is lossless.

Checking R1 ⋈ R2. R1 ∩ R2 = {∅}. As the intersection of R1 and R2 is a null set, the decomposition is lossy.

To check R13 and R2
R13 = {A, B, C, D} and R2. R13 ∩ R2 = {D}
{D}+ = {D}. We can see that D is not a candidate key of either R13 or R2. So, the decomposition R13 and R2 is lossy.

Now, we check R2 ⋈ R3. R2 ∩ R3 = {D}.
{D}+ = {D}. Thus, we can see that D is not a candidate key of either R3 or R2. Hence the decomposition R2 and R3 is lossy.

Since all individual decompositions are lossy, the decomposition R1, R2 and R3 is lossy.

Question assigned to the following page: 2

Part 2

Entities:

1. Person(Person_ID, Name, DOB, Gender)
   Non-null attributes: Person_ID, Name, DOB, Gender
   Key: Person_ID
   IND: None

2. Employee(Person_ID, Schedule, Employee_Type, Salary_per_hour)
   Non-null attributes: Person_ID, Schedule, Employee_Type, Salary_per_hour
   Key: Person_ID
   IND: Employee[Person_ID] ⊆ Person[Person_ID]

3. Entry_Log(Person_ID, Timestamp)
   Non-null attributes: Person_ID, Timestamp
   Key: Person_ID, Timestamp
   IND: Entry_Log[Person_ID] ⊆ Person[Person_ID]

4. Employee_Exit_log(Person_ID, Timestamp)
   Non-null attributes: Person_ID, Timestamp
   Key of Entry_Log: Person_ID, Timestamp
   IND: Employee_Exit_log[Person_ID] ⊆ Employee[Person_ID]

5. Trainer(Person_ID, Credentials)
   Non null attributes: Person_ID, Credentials
   Key: Person_ID
   IND: Trainer[Person_ID] ⊆ Employee[Person_ID]

6. DeskEmployee(Person_ID)
   Non null attributes: Person_ID
   Key: Person_ID
   IND: DeskEmployee[Person_ID] ⊆ Employee[Person_ID]

7. Member(Person_ID, Membership_ID)
   Non null attributes: Person_ID, Membership_ID
   Key: Person_ID
   IND: Member[Person_ID] ⊆ Person[Person_ID]

8. Family(Person_ID, CreditCard)
   Non null attributes: Person_ID, CreditCard
   Key: Person_ID
   IND: Family[Person_ID] ⊆ Member[Person_ID]

Question assigned to the following page: 2

9. University_Affiliate(Person_ID, Department)
   Non null attributes: Person_ID, Department
   Key: Person_ID
   IND: University_Affiliate[Person_ID] ⊆ Member[Person_ID]

10. Student(Person_ID, Student_Type)
    Non null attributes: Person_ID, Student_Type
    Key: Person_ID
    IND: Student[Person_ID] ⊆ University_Affiliate[Person_ID]

11. Non-Student(Person_ID, Member_type, Credit_card)
    Non null attributes: Person_ID, Member_type, Credit_card
    Key: Person_ID
    IND: Non-Student[Person_ID] ⊆ University_Affiliate[Person_ID]

12. Space(Space_ID, Description, Max_Capacity)
    Non-null attributes: Space_ID, Description, Max_Capacity
    Key: Space_ID
    IND (INclusion Dependencies): None

13. Events(Event_ID, Description, Start_time, end_time, capacity)
    Non-null attributes: Event_ID, Description, Start_time, end_time, capacity
    Key: Event_ID
    IND (INclusion Dependencies): None

14. Location_Reading(Person_ID, Space_ID, Sensor_ID, Timestamp)
    Key: Person_ID, Space_ID, Sensor_ID, Timestamp
    Non-null attributes: Person_ID, Space_ID, Sensor_ID, Timestamp
    IND : Location_Reading[Person_ID] ⊆ Person[Person_ID]
          Location_Reading[Space_ID] ⊆ Space[Space_ID]
          Location_Reading[Sensor_ID] ⊆ LocationSensor[Sensor_ID]

15. Equipment(Equipment_ID, Equipment_type, is_available)
    Key: Equipment_ID
    Non-null attributes: Equipment_ID, Equipment_type, is_available
    IND (INclusion Dependencies): None

16. LocationSensor(Sensor_ID, Coverage)
    Non-null attributes: Sensor_ID, Coverage

Question assigned to the following page: 2

Key: Sensor_ID
IND (INclusion Dependencies): None

17. EquipmentSensor(Sensor_ID, Coverage)
    Non-null attributes: Sensor_ID, Coverage
    Key: Sensor_ID
    IND (INclusion Dependencies): None

18. UsageReading(Equipment_ID, Person_ID, timestamp, Sensor_ID)
    Key of UsageReading: Equipment_ID, Person_ID, timestamp, Sensor_ID
    Non-null attributes: Equipment_ID, Person_ID, timestamp, Sensor_ID
    IND: UsageReading[Equipment_ID] ⊆ Equipment[Equipment_ID]
    UsageReading[Person_ID] ⊆ Person[Person_ID]
    UsageReading[Sensor_ID] ⊆ EquipmentSensor[Sensor_ID]

Relations:

1. MemberAttendsEvents(Person_ID, Event_ID)
   Key: Person_ID, Event_ID
   Non-null attributes: Person_ID, Event_ID
   IND (INclusion Dependencies):
   MemberAttendsEvents[Person_ID] ⊆ Member[Person_ID]
   MemberAttendsEvents[Event_ID] ⊆ Event[Event_ID]

2. EventsHostedIn(EventId, Space_Id)
   Key: Event_id
   Non-null attributes: Event_id, Space_Id
   IND:
   EventsHostedIn[Event_Id] ⊆ Event[Event_Id]
   Event[Event_Id] ⊆ EventsHostedIn[Event_Id]
   EventsHostedIn[Space_ID] ⊆ Space[Space_ID]

3. SpaceContainsEquipments(Equipment_Id, Space_Id)
   Key: Equipment_Id
   Non-null attributes: Equipment_Id, Space_Id
   IND:
   SpaceContainsEquipments[Equipment_Id] ⊆ Equipment[Equipment_Id]
   Equipment[Equipment_Id] ⊆ SpaceContainsEquipments[Equipment_Id]
   SpaceContainsEquipments[Space_ID] ⊆ Space[Space_ID]

4. FamilyRealtedTo(FamilyPersonId, UniversityAffiliatePersonId)
   Key: FamilyPersonId
   Non-null attributes: FamilyPersonId, UniversityAffiliatePersonId

Question assigned to the following page: 2

IND:

FamilyRealtedTo[FamilyPersonId] ⊆ Family[Person_ID]
Family[Person_ID] ⊆ FamilyRealtedTo[FamilyPersonId]
FamilyRealtedTo[UniversityAffiliatePersonId] ⊆ University_Affiliate[Person_ID]

Queries:

Entities

```
CREATE TABLE Person (
   Person_ID INT PRIMARY KEY,
   Name VARCHAR(255) NOT NULL,
   DOB DATE NOT NULL,
   Gender VARCHAR(10) NOT NULL
);

CREATE TABLE Employee (
   Person_ID INT PRIMARY KEY,
   Schedule VARCHAR(255) NOT NULL,
   Employee_Type VARCHAR(255) NOT NULL,
   Salary_per_hour DECIMAL(10,2) NOT NULL,
   FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID)
);
```

Question assigned to the following page: 2

```
CREATE TABLE Entry_Log (
    Person_ID INT,
    Timestamp TIMESTAMP NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID)
);

CREATE TABLE Employee_Exit_Log (
    Person_ID INT,
    Timestamp TIMESTAMP NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Employee(Person_ID)
);

CREATE TABLE Trainer (
    Person_ID INT PRIMARY KEY,
    Credentials VARCHAR(255) NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Employee(Person_ID)
);

CREATE TABLE DeskEmployee (
    Person_ID INT PRIMARY KEY,
    FOREIGN KEY (Person_ID) REFERENCES Employee(Person_ID)
);

CREATE TABLE Member (
    Person_ID INT PRIMARY KEY,
    Membership_ID INT NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID)
);

CREATE TABLE Family (
    Person_ID INT PRIMARY KEY,
    CreditCard VARCHAR(255) NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Member(Person_ID)
);

CREATE TABLE University_Affiliate (
    Person_ID INT PRIMARY KEY,
    Department VARCHAR(255) NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Member(Person_ID)
);

CREATE TABLE Student (
    Person_ID INT PRIMARY KEY,
    Student_Type VARCHAR(255) NOT NULL,
```

Question assigned to the following page: 2

```
    FOREIGN KEY (Person_ID) REFERENCES University_Affiliate(Person_ID)
);

CREATE TABLE Non_Student (
    Person_ID INT PRIMARY KEY,
    Member_type VARCHAR(255) NOT NULL,
    Credit_card VARCHAR(255) NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES University_Affiliate(Person_ID)
);

CREATE TABLE Space (
    Space_ID INT PRIMARY KEY,
    Description VARCHAR(255) NOT NULL,
    Max_Capacity INT NOT NULL
);

CREATE TABLE Events (
    Event_ID INT PRIMARY KEY,
    Description VARCHAR(255) NOT NULL,
    Start_time DATETIME NOT NULL,
    End_time DATETIME NOT NULL,
    Capacity INT NOT NULL
);

CREATE TABLE LocationSensor (
    Sensor_ID INT PRIMARY KEY,
    Coverage VARCHAR(255) NOT NULL
);

CREATE TABLE Location_Reading (
    Person_ID INT,
    Space_ID INT,
    Sensor_ID INT,
    Timestamp TIMESTAMP NOT NULL,
    FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID),
    FOREIGN KEY (Space_ID) REFERENCES Space(Space_ID),
    FOREIGN KEY (Sensor_ID) REFERENCES LocationSensor(Sensor_ID)
);

CREATE TABLE Equipment (
    Equipment_ID INT PRIMARY KEY,
    Equipment_type VARCHAR(255) NOT NULL,
    is_available BOOLEAN NOT NULL
);
```

Question assigned to the following page: 2

```
CREATE TABLE EquipmentSensor (
    Sensor_ID INT PRIMARY KEY,
    Coverage VARCHAR(255) NOT NULL
);

CREATE TABLE UsageReading (
    Equipment_ID INT,
    Person_ID INT,
    Timestamp TIMESTAMP NOT NULL,
    Sensor_ID INT,
    FOREIGN KEY (Equipment_ID) REFERENCES Equipment(Equipment_ID),
    FOREIGN KEY (Person_ID) REFERENCES Person(Person_ID),
    FOREIGN KEY (Sensor_ID) REFERENCES EquipmentSensor(Sensor_ID)
);
```

Relations

```
CREATE TABLE MemberAttendsEvents (
    Person_ID INT,
    Event_ID INT,
    PRIMARY KEY (Person_ID, Event_ID),
    FOREIGN KEY (Person_ID) REFERENCES Member(Person_ID),
    FOREIGN KEY (Event_ID) REFERENCES Events(Event_ID)
);

CREATE TABLE EventsHostedIn (
    Event_ID INT PRIMARY KEY,
    Space_ID INT,
    FOREIGN KEY (Event_ID) REFERENCES Events(Event_ID),
    FOREIGN KEY (Space_ID) REFERENCES Space(Space_ID)
);

CREATE TABLE SpaceContainsEquipments (
    Equipment_ID INT PRIMARY KEY,
    Space_ID INT,
    FOREIGN KEY (Equipment_ID) REFERENCES Equipment(Equipment_ID),
    FOREIGN KEY (Space_ID) REFERENCES Space(Space_ID)
);

CREATE TABLE FamilyRealtedTo (
```

Question assigned to the following page: 2

```
    FamilyPersonId INT PRIMARY KEY,
    UniversityAffiliatePersonId INT,
    FOREIGN KEY (FamilyPersonId) REFERENCES Family(Person_ID),
    FOREIGN KEY (UniversityAffiliatePersonId) REFERENCES
University_Affiliate(Person_ID)
);
```