29771457—SQL Programming & Creative Writing

Task 1

- Transaction is an action or series of actions the execution of which should lead to a consistent database state from another consistent database state. Discuss which properties that transactions should have for their correct executions. Provide two examples to support your answer.

> *Cite any source in APA format.*

Database transactions are expected to display "atomicity, consistency, isolation, and durability" (Rob & Coronel, 2009, pg. 417). The atomicity property demands that each and every constituent SQL query or SQL operation of a transaction must complete successfully for the transaction itself to complete. If this does not happen, then the given transaction should abort. An example is when you have a relation named `Specialty` that exists in a particular database. If the relation has two attributes named `SpecialtyID` and `SpecialtyName`; a transaction which seeks to insert values into the relation could look like:

```
INSERT INTO Specialty VALUES
    ('S1', 'Dermatology'),
    ('S2', Psychiatry),
    ('S3', 'Oncology');
```

The first and third set of values suggest that the `SpecialtyID` and `SpecialtyName` columns accept `String` values only. Yet, the second set of values contains an unquoted `String` value which it seeks to insert into the `SpecialtyName` column. Because this SQL statement constitutes a transaction, its execution would fail with a message such as:

```
ERROR 1054 (42S22): Unknown column 'Psychiatry' in 'field list'
```

Hence, even though the first and third set of values were correct, they too would not be executed because of the failing part of the statement.

On the other hand, the consistency property demands that a transaction should not violate an integrity constraint even if its constituent SQL statements can complete successfully on their own. That is, a transaction should leave a database in its consistent state. Thus, if you have a relation named `DancingPony` with two attributes that make up its primary key (which should not be `NULL`) named `DanceID` and `PonyID`—any given transaction whose statement attempts to insert `NULL` into any of the attributes should abort.

Reference

Rob, P., & Coronel, C. (2009). *Database systems: Design, implementation, and management*. Thomson/Course Technology.

Task 2

- Using the `Specialty`, `Doctor`, `Patient`, `Appointment`, `Allergy`, `PatientAllergy`, `Medicine`, and `PatientMedicine` relations provided in the attached Task 2 SQL code, populate them with data using the the provided tables in the attached Task 2. If the relations in the provided SQL code have additional attributes that are not included in the provided tables, then add appropriate values to populate your relations with data.

Assignment instructions

- *Provide all of the SQL statements required to create the relations.*

```
CREATE TABLE Specialty (
    SpecialtyNumber VARCHAR(4) NOT NULL,
    SpecialtyName VARCHAR(45) NOT NULL,
    PRIMARY KEY (SpecialtyNumber)
);

CREATE TABLE Doctor (
    DoctorID VARCHAR(4) NOT NULL,
    Name VARCHAR(155) NOT NULL,
    Phone VARCHAR(20) NOT NULL,
    SpecialtyNumber VARCHAR(4) NOT NULL,
```

```
    Supervisor VARCHAR(4) DEFAULT NULL,
    PRIMARY KEY (DoctorID),
    CONSTRAINT fk_doctor_specialty_number FOREIGN KEY
    (SpecialtyNumber) REFERENCES Specialty (SpecialtyNumber),
    CONSTRAINT fk_doctor_supervisor FOREIGN KEY
    (Supervisor) REFERENCES Doctor (DoctorID)
);

CREATE TABLE Patient (
    PatientID VARCHAR(4) NOT NULL,
    DoctorID VARCHAR(4) NOT NULL,
    Name VARCHAR(155) NOT NULL,
    Phone VARCHAR(20) NOT NULL,
    Email VARCHAR(50) DEFAULT NULL,
    Address VARCHAR(50) DEFAULT NULL,
    AddedDate DATE NOT NULL DEFAULT (CURRENT_DATE),
    PRIMARY KEY (PatientID),
    CONSTRAINT fk_patient_doctor FOREIGN KEY (DoctorID)
    REFERENCES Doctor (DoctorID)
);

CREATE TABLE Appointment (
    AppointmentID VARCHAR(4) NOT NULL,
    AppointmentDate DATE NOT NULL,
    -- A BP reading should be entered as: "132/88";
    -- where the units are "mmHg"
    BloodPressure VARCHAR(7) NOT NULL,
    -- Pulse should be entered as: 90;
    -- where the units are "BPM"
    -- Pulse should not be negative, hence is unsigned
    Pulse SMALLINT UNSIGNED NOT NULL,
    -- Weight should not be negative, hence is unsigned
    Weight SMALLINT UNSIGNED NOT NULL,
    TreatmentNotes VARCHAR(255) NOT NULL,
    PRIMARY KEY (AppointmentID, AppointmentDate)
);

CREATE TABLE PatientAppointment (
    PatientID VARCHAR(4) NOT NULL,
    AppointmentID VARCHAR(4) NOT NULL,
    PRIMARY KEY (PatientID, AppointmentID),
    CONSTRAINT fk_patient_appointment_patient FOREIGN KEY
    (PatientID) REFERENCES Patient (PatientID),
    CONSTRAINT fk_patient_appointment_appointment FOREIGN KEY
    (AppointmentID) REFERENCES Appointment (AppointmentID)
);

CREATE VIEW Appointments AS
    SELECT
        PatientAppointment.AppointmentID,
        PatientAppointment.PatientID,
        Patient.DoctorID,
        Appointment.AppointmentDate,
        Appointment.BloodPressure,
        Appointment.Pulse,
        Appointment.Weight,
        Appointment.TreatmentNotes
    FROM (Appointment
            LEFT JOIN PatientAppointment
                ON Appointment.AppointmentID = PatientAppointment.AppointmentID
            LEFT JOIN Patient
                ON PatientAppointment.PatientID = Patient.PatientID);

CREATE TABLE Allergy (
    AllergyID VARCHAR(4) NOT NULL,
    AllergyName VARCHAR(155) NOT NULL,
    PRIMARY KEY (AllergyID)
);

CREATE TABLE PatientAllergy (
    PatientID VARCHAR(4) NOT NULL,
```

```
    AllergyID VARCHAR(4) NOT NULL,
    PRIMARY KEY (PatientID, AllergyID),
    CONSTRAINT fk_patient_allergy_patient FOREIGN KEY (PatientID)
    REFERENCES Patient (PatientID),
    CONSTRAINT fk_patient_allergy_allergy FOREIGN KEY (AllergyID)
    REFERENCES Allergy (AllergyID)
);

CREATE TABLE Medicine (
    MedicineID VARCHAR(4) NOT NULL,
    MedicineName VARCHAR(155) NOT NULL,
    PRIMARY KEY (MedicineID)
);

CREATE TABLE AppointmentMedicine (
    AppointmentID VARCHAR(4) NOT NULL,
    MedicineID VARCHAR(4) NOT NULL,
    PRIMARY KEY (AppointmentID, MedicineID),
    CONSTRAINT fk_appointment_medicine_appointment
    FOREIGN KEY (AppointmentID)
    REFERENCES Appointment (AppointmentID),
    CONSTRAINT fk_appointment_medicine_medicine
    FOREIGN KEY (MedicineID)
    REFERENCES Medicine (MedicineID)
);

CREATE VIEW PatientMedicine AS
    SELECT AppointmentID, MedicineID
    FROM AppointmentMedicine;
```

- *Populate the relations with data (using SQL insert statements) by using information in the provided tables.*

```
INSERT INTO Specialty VALUES
    ('S1', 'Dermatology'),
    ('S2', 'Psychiatry'),
    ('S3', 'Oncology'),
    ('S4', 'Cardiology'),
    ('S5', 'Urology'),
    ('S6', 'Pediatrics');

INSERT INTO Doctor
    (DoctorID, Name, Phone, SpecialtyNumber)
    VALUES
    ('D1', 'Doctor Karen', '555-1212', 'S6');

INSERT INTO Doctor VALUES
    ('D2', 'Doctor John', '555-2934', 'S2', 'D1'),
    ('D3', 'Doctor Robert', '555-6723', 'S6', 'D1'),
    ('D4', 'Doctor David', '555-1745', 'S4', 'D1'),
    ('D5', 'Doctor Mary', '555-6565', 'S5', 'D1'),
    ('D6', 'Doctor Linda', '555-4889', 'S1', 'D1'),
    ('D7', 'Doctor Susan', '555-4581', 'S3', 'D1'),
    ('D8', 'Doctor Zeynep', '555-7891', 'S4', 'D1'),
    ('D9', 'Doctor Mat', '555-7791', 'S1', 'D1');

INSERT INTO Patient VALUES
    ('P1', 'D2', 'Patient Dana', '444-1212', 'P1@email.com', '123 Home St.', '2019-02-01'),
    ('P2', 'D7', 'Patient Harry', '444-2934', 'P2@email.com', '3435 Main St.', '2011-07-13'),
    ('P3', 'D6', 'Patient Karl', '444-6723', 'P3@email.com', '2176 Baker St.', '2009-05-10'),
    ('P4', 'D2', 'Patient Sid', '444-1745', 'P4@email.com', '176 Right St.', '2010-06-20'),
    ('P5', 'D8', 'Patient Marry', '444-6565', 'P5@email.com', '435 Main St.', '2014-05-18'),
    ('P6', 'D6', 'Patient Kim', '444-4889', 'P6@email.com', '34 Home St.', '2018-03-15'),
    ('P7', 'D4', 'Patient Susan', '444-4581', 'P7@email.com', '65 Water St.', '2011-09-07'),
    ('P8', 'D3', 'Patient Sam', '444-7891', 'P8@email.com', '23 Hill Drive', '2010-11-23'),
    ('P9', 'D5', 'Patient Peter', '444-7791', 'P9@email.com', '12 River St.', '2008-02-01'),
    ('P10', 'D7', 'Patient Nick', '123-1212', 'P10@email.com', '335 Bay St.', '2011-07-13'),
    ('P11', 'D9', 'Patient Kyle', '123-2934', 'P11@email.com', '216 Baker St.', '2016-05-10'),
    ('P12', 'D9', 'Patient Garcia', '123-6723', 'P12@email.com', '176 Right St.', '2010-06-20'),
    ('P13', 'D4', 'Patient Alicia', '123-1745', 'P13@email.com', '823 Left St.', '2015-05-18'),
    ('P14', 'D4', 'Patient Dan', '123-6565', 'P14@email.com', '534 High St.', '2018-03-15');
```

```sql
INSERT INTO Appointment VALUES
    ('A1', '2019-07-01', '177/114', 80, 65, 'Dream to success'),
    ('A2', '2019-01-04', '161/113', 77, 88, 'Good heart rate'),
    ('A3', '2019-03-22', '155/112', 82, 95, 'Many spots'),
    ('A4', '2020-02-01', '135/100', 85, 74, 'Fast heart rate'),
    ('A5', '2019-04-13', '161/92', 75, 56, 'Reports checked'),
    ('A6', '2019-11-12', '175/84', 81, 96, 'Sun light spots'),
    ('A7', '2020-01-29', '138/109', 80, 87, 'Early treatment'),
    ('A8', '2019-08-12', '156/102', 86, 92, 'Much better'),
    ('A9', '2019-05-18', '149/117', 75, 75, 'Good heart rate'),
    ('A10', '2019-11-18', '176/115', 76, 79, 'New teeth'),
    ('A11', '2019-06-22', '163/81', 78, 71, 'Much better'),
    ('A12', '2020-02-21', '132/114', 82, 86, 'Early treatment'),
    ('A13', '2019-08-17', '172/97', 81, 101, 'Bad dreams'),
    ('A14', '2019-06-27', '154/120', 79, 49, 'Sun light spots'),
    ('A15', '2020-07-29', '125/84', 80, 83, 'Early treatment'),
    ('A16', '2020-08-01', '135/118', 78, 79, 'Good heart rate');

INSERT INTO PatientAppointment VALUES
    ('P1', 'A1'),
    ('P13', 'A2'),
    ('P11', 'A3'),
    ('P7', 'A4'),
    ('P9', 'A5'),
    ('P3', 'A6'),
    ('P10', 'A7'),
    ('P9', 'A8'),
    ('P14', 'A9'),
    ('P8', 'A10'),
    ('P11', 'A11'),
    ('P2', 'A12'),
    ('P4', 'A13'),
    ('P6', 'A14'),
    ('P10', 'A15'),
    ('P7', 'A16');

INSERT INTO Allergy VALUES
    ('AL1', 'Drug'),
    ('AL2', 'Food'),
    ('AL3', 'Skin'),
    ('AL4', 'Asthma'),
    ('AL5', 'Rhinitis');

INSERT INTO PatientAllergy (AllergyID, PatientID) VALUES
    ('AL4', 'P1'),
    ('AL2', 'P13'),
    ('AL3', 'P11'),
    ('AL4', 'P7'),
    ('AL5', 'P9'),
    ('AL1', 'P3');

INSERT INTO Medicine VALUES
    ('M1', 'Ativan'),
    ('M2', 'Ibuprofen'),
    ('M3', 'Omeprazole'),
    ('M4', 'Metoprolol'),
    ('M5', 'Azithromycin'),
    ('M6', 'Codeine');

INSERT INTO AppointmentMedicine VALUES
    ('A15', 'M1'),
    ('A2', 'M6'),
    ('A8', 'M3'),
    ('A6', 'M3'),
    ('A15', 'M2'),
    ('A10', 'M6'),
    ('A10', 'M2'),
    ('A4', 'M5'),
    ('A3', 'M5'),
    ('A1', 'M2');
```

- *Issue a select statement for each relation to retrieve data in relations.*

```sql
SELECT * FROM Specialty;
SELECT * FROM Doctor;
SELECT * FROM Patient;
SELECT * FROM Appointments;
SELECT * FROM Allergy;
SELECT AllergyID, PatientID FROM PatientAllergy;
SELECT * FROM Medicine;
SELECT * FROM PatientMedicine;
```

- *Include both the select statement and the output of the select statement (using a screenshot of your database's output) that shows the contents of each relation.*

Specialty

```
MariaDB [29771457_task2]> SELECT * FROM Specialty;
+-----------------+---------------+
| SpecialtyNumber | SpecialtyName |
+-----------------+---------------+
| S1              | Dermatology   |
| S2              | Psychiatry    |
| S3              | Oncology      |
| S4              | Cardiology    |
| S5              | Urology       |
| S6              | Pediatrics    |
+-----------------+---------------+
6 rows in set (0.000 sec)
```

Doctor

```
MariaDB [29771457_task2]> SELECT * FROM Doctor;
+----------+--------------+----------+-----------------+------------+
| DoctorID | Name         | Phone    | SpecialtyNumber | Supervisor |
+----------+--------------+----------+-----------------+------------+
| D1       | Doctor Karen | 555-1212 | S6              | NULL       |
| D2       | Doctor John  | 555-2934 | S2              | D1         |
| D3       | Doctor Robert| 555-6723 | S6              | D1         |
| D4       | Doctor David | 555-1745 | S4              | D1         |
| D5       | Doctor Mary  | 555-6565 | S5              | D1         |
| D6       | Doctor Linda | 555-4889 | S1              | D1         |
| D7       | Doctor Susan | 555-4581 | S3              | D1         |
| D8       | Doctor Zeynep| 555-7891 | S4              | D1         |
| D9       | Doctor Mat   | 555-7791 | S1              | D1         |
+----------+--------------+----------+-----------------+------------+
9 rows in set (0.000 sec)
```

Patient

```
MariaDB [29771457_task2]> SELECT * FROM Patient;
+-----------+----------+---------------+----------+---------------+----------------+------------+
| PatientID | DoctorID | Name          | Phone    | Email         | Address        | AddedDate  |
+-----------+----------+---------------+----------+---------------+----------------+------------+
| P1        | D2       | Patient Dana  | 444-1212 | P1@email.com  | 123 Home St.   | 2019-02-01 |
| P10       | D7       | Patient Nick  | 123-1212 | P10@email.com | 335 Bay St.    | 2011-07-13 |
| P11       | D9       | Patient Kyle  | 123-2934 | P11@email.com | 216 Baker St.  | 2016-05-10 |
| P12       | D9       | Patient Garcia| 123-6723 | P12@email.com | 176 Right St.  | 2010-06-20 |
| P13       | D4       | Patient Alicia| 123-1745 | P13@email.com | 823 Left St.   | 2015-05-18 |
| P14       | D4       | Patient Dan   | 123-6565 | P14@email.com | 534 High St.   | 2018-03-15 |
| P2        | D7       | Patient Harry | 444-2934 | P2@email.com  | 3435 Main St.  | 2011-07-13 |
| P3        | D6       | Patient Karl  | 444-6723 | P3@email.com  | 2176 Baker St. | 2009-05-10 |
| P4        | D2       | Patient Sid   | 444-1745 | P4@email.com  | 176 Right St.  | 2010-06-20 |
| P5        | D8       | Patient Marry | 444-6565 | P5@email.com  | 435 Main St.   | 2014-05-18 |
| P6        | D6       | Patient Kim   | 444-4889 | P6@email.com  | 34 Home St.    | 2018-03-15 |
| P7        | D4       | Patient Susan | 444-4581 | P7@email.com  | 65 Water St.   | 2011-09-07 |
| P8        | D3       | Patient Sam   | 444-7891 | P8@email.com  | 23 Hill Drive  | 2010-11-23 |
| P9        | D5       | Patient Peter | 444-7791 | P9@email.com  | 12 River St.   | 2008-02-01 |
+-----------+----------+---------------+----------+---------------+----------------+------------+
14 rows in set (0.001 sec)
```

Appointments

```
MariaDB [29771457_task2]> SELECT * FROM Appointments;
+---------------+-----------+----------+-----------------+---------------+-------+--------+------------------+
| AppointmentID | PatientID | DoctorID | AppointmentDate | BloodPressure | Pulse | Weight | TreatmentNotes   |
+---------------+-----------+----------+-----------------+---------------+-------+--------+------------------+
| A1            | P1        | D2       | 2019-07-01      | 177/114       |    80 |     65 | Dream to success |
| A10           | P8        | D3       | 2019-11-18      | 176/115       |    76 |     79 | New teeth        |
| A11           | P11       | D9       | 2019-06-22      | 163/81        |    78 |     71 | Much better      |
| A12           | P2        | D7       | 2020-02-21      | 132/114       |    82 |     86 | Early treatment  |
| A13           | P4        | D2       | 2019-08-17      | 172/97        |    81 |    101 | Bad dreams       |
| A14           | P6        | D6       | 2019-06-27      | 154/120       |    79 |     49 | Sun light spots  |
| A15           | P10       | D7       | 2020-07-29      | 125/84        |    80 |     83 | Early treatment  |
| A16           | P7        | D4       | 2020-08-01      | 135/118       |    78 |     79 | Good heart rate  |
| A2            | P13       | D4       | 2019-01-04      | 161/113       |    77 |     88 | Good heart rate  |
| A3            | P11       | D9       | 2019-03-22      | 155/112       |    82 |     95 | Many spots       |
| A4            | P7        | D4       | 2020-02-01      | 135/100       |    85 |     74 | Fast heart rate  |
| A5            | P9        | D5       | 2019-04-13      | 161/92        |    75 |     56 | Reports checked  |
| A6            | P3        | D6       | 2019-11-12      | 175/84        |    81 |     96 | Sun light spots  |
| A7            | P10       | D7       | 2020-01-29      | 138/109       |    80 |     87 | Early treatment  |
| A8            | P9        | D5       | 2019-08-12      | 156/102       |    86 |     92 | Much better      |
| A9            | P14       | D4       | 2019-05-18      | 149/117       |    75 |     75 | Good heart rate  |
+---------------+-----------+----------+-----------------+---------------+-------+--------+------------------+
16 rows in set (0.001 sec)
```

Allergy

```
MariaDB [29771457_task2]> SELECT * FROM Allergy;
+-----------+-------------+
| AllergyID | AllergyName |
+-----------+-------------+
| AL1       | Drug        |
| AL2       | Food        |
| AL3       | Skin        |
| AL4       | Asthma      |
| AL5       | Rhinitis    |
+-----------+-------------+
5 rows in set (0.001 sec)
```

PatientAllergy

```
MariaDB [29771457_task2]> SELECT AllergyID, PatientID FROM PatientAllergy;
+-----------+-----------+
| AllergyID | PatientID |
+-----------+-----------+
| AL4       | P1        |
| AL3       | P11       |
| AL2       | P13       |
| AL1       | P3        |
| AL4       | P7        |
| AL5       | P9        |
+-----------+-----------+
6 rows in set (0.001 sec)
```

Medicine

```
MariaDB [29771457_task2]> SELECT * FROM Medicine;
+------------+--------------+
| MedicineID | MedicineName |
+------------+--------------+
| M1         | Ativan       |
| M2         | Ibuprofen    |
| M3         | Omeprazole   |
| M4         | Metoprolol   |
| M5         | Azithromycin |
| M6         | Codeine      |
+------------+--------------+
6 rows in set (0.001 sec)
```

PatientMedicine

```
MariaDB [29771457_task2]> SELECT * FROM PatientMedicine;
+---------------+------------+
| AppointmentID | MedicineID |
+---------------+------------+
| A1            | M2         |
| A10           | M2         |
| A10           | M6         |
| A15           | M1         |
| A15           | M2         |
| A2            | M6         |
| A3            | M5         |
| A4            | M5         |
| A6            | M3         |
| A8            | M3         |
+---------------+------------+
10 rows in set (0.001 sec)
```

Task 3

Based on Task 2, answer the following using Introduction-Body-Conclusion format:

- Describe what you did (This does not mean that you copy and paste from what you have posted or the assignments you have prepared. You need to describe what you did and how you did it), what you learned, your weekly activities, in what ways are you able to apply the ideas and concepts gained, and finally, describe one important thing that you are thinking about in relation to the activity.
- List two main approaches to database programming and discuss the advantages and disadvantages of each approach.

> *Your response should be at least **400 words** in length, but not more than 750 words. Cite any source in APA format.*

To complete this task, I had several SQL `CREATE` statements and tables displaying the values that the resultant relations should contain. Yet, there were a few discrepancies between what the SQL statements could achieve and what the provided values suggested. One notable example is the `Appointments` relation which was displayed in a table that contained columns from other relations. This meant that if I were to match what was in the `Appointments` table, I would have to create a view which pulled data from other relations by using the `JOIN` operator. This was also true for another relatively simple table named `PatientMedicine`. Even though it contained columns from one relation named `AppointmentMedicine`, it too demanded the creation of a view. However, this view did not necessitate the use of a `JOIN` operator because a simple `SELECT` operation worked just as well.

These activities came just at the right time because they built upon the weekly tasks of learning how to use the `JOIN` operator. They were also well timed because they took place after I had learned how to determine how to choose between using views and tables only when seeking to produce value outputs in a particular way. Yet, as I was combining the use of table queries and views, I wondered whether I could always be proficient in such tasks in the absence of test data. The task's requirements, for instance, came with test data. However, such a provision suggested to me that I should always create test data myself whenever I dealt with tables from several relations. The problem with this is that such data could be bothersome to produce when it is needed in significant quantities.

Database programming is meant to interact closely with general-purpose programming (Elmasri & Navathe, 2016). Yet, the two feature disparate syntax and compiling facilities. Still, the interaction is made possible by having SQL statements that can be embedded in the host programming language. In C programming, we have embedded SQL; and in Java we have SQLJ. This phenomenon simplifies the development process because a developer gets to use one editor for the entire codebase. Yet, the way the database programming syntax is embedded in a language such as Java is purely through the use of `String` objects which are then passed to the host language's objects. This creates an increased burden on the developer when debugging in case of typos in the embedded SQL, for instance. On the other hand, database programming necessitates the use of class libraries to manage database calls and functions from within the host language. Java, for instance, has its JDBC class library that creates a bridge between Java code and the underlying database systems. This is advantageous because such class libraries can work with nearly all database systems that are in the market. Yet, adapting the calls to such class libraries to correct abstraction principles proves challenging in most cases. As a result, it is not unusual to find low level JDBC calls in other parts of a modular program —like in the UI. This has the effect of creating tight coupling between the layers of a program. Unfortunately in agile programming environments this forces some developers to work across many modules; thus, making the development process inefficient.

Reference

Elmasri, R., & Navathe, S. (2016). *Fundamentals of database systems*. Pearson.