CPSC 368: Databases in Data Science
Practice Exercises: Writing SQL

Consider the following relational schema for an orchestra.

1. Person(email, name, age)
    - This relation stores anyone who has signed up for our mailing list. Tuples in this relation may not be listed in Purchase.
    - The minimum age for someone to sign up for the mailing list is 11.
2. Show(id, year, month, date, showing, attendanceNumber)
    - Showing describes whether a show was during morning, afternoon, or evening
3. Song(composer, title)
4. SongsPerformed(**showID**, **composer**, **title**)
    - showID is a foreign key referring to Show
    - composer and title are foreign keys referring to attributes of the same name in Song
5. Purchase(**email**, **showID**, price)
    - email is a foreign key referring to the email attribute in Person
    - showID is a foreign key referring to Show
6. SymphonyEmployee(id, name, nationality)
7. Musician(**eID**, name, country)
    - eID refers to id in SymphonyEmployee
8. ConcessionsEmployee(**eID**, standNumber)
    - eID refers to id in SymphonyEmployee
9. PerformedIn(**id**, **showID**)
    - id refers to the attribute of the same name in Musician
    - showID is a foreign key referring to Show
10. Merchandise(id, itemType, description, price)
11. MerchSales(**eID**, **showID**, **merchandiseID**, creditCardNumber, **email**)
    - eID refers to id in SymphonyEmployee
    - showID refers to the id attribute in Show
    - merchandiseID refers to the id attribute in Merchandise
    - email refers to the email attribute in Person

Part 1: Using GROUP BY

1. Find all the months and years where the total sum of sales for that month exceeds $2000. For example, if the total sum of sales for November 2019 is 1800 and the total um of sales for December 2019 is 5000, your query should return “December, 2019” as a tuple in the set of results.
2. Find the email addresses of people who have attended at least two shows.
3. Find the merchandise item types where the total sales of that item type exceed $500?
4. Find all the instruments with more than two musicians who play it.

Part 2: Aggregate Operators

1. Find the musicians who are younger than the average age of the all the musicians in the orchestra.
2. Find the total number of attendees for each day there was a show. Don’t forget that a single day can have multiple shows!
3. Find the total number of Canadian musicians per instrument.
4. For all the shows where the audience’s average age is greater than the average age of our mailing list, what songs were performed?

Part 3: Updating Data

1. The orchestra has decided to increase the price of any merchandise item with the word "book" in its description. Specifically, the price will increase by $10.

   Write an UPDATE statement that would accomplish this task.
2. As the economy has gone into a recession, the orchestra has decided to lower the cost of any piece of merchandise with the word "book" in its description. Each of those items will have its price lowered by 10. The minimum price a produce can be is 1.

   Write an UPDATE statement that would accomplish this task.
3. Due to a system glitch, the information about people on our mailing list has been stored incorrectly. Specifically, the ages of people who have a name that starts with the letter "K" is incorrect. Everyone is recorded as 2 years older than they actually are.

   Write an UPDATE statement that would update our mailing list so that everyone has the correct age.
4. Due to a system glitch, the information about people on our mailing list has been stored incorrectly. Specifically, the ages of people who have an "A" as the second letter in their name is incorrect. Everyone is recorded as 5 years younger than they actually are.

   Write an UPDATE statement that would update our mailing list so that everyone has the correct age.

Part 4: Division (Method 1 – i.e., with EXCEPT)

1. What songs have been performed by every violinist in the orchestra?
2. Which composers has every attendee under the age of 30 listened to?
3. Find the pieces of merchandise that were purchased by every person under 30.
4. Find the email addresses of people who have purchased a ticket for every Tchaikovsky performance.
5. Find the musicians who have performed in every showing held during January 2020.

Part 5: Practice with Views

Try to make use of views in your solution.

1. Who were the musicians (just find the musician IDs) who performed in the lowest attended shows for January 2020 and February 2020?
2. Which merchandise items were sold during the lowest attended shows for January 2020 and February 2020?

Answer Key

Part 1

1. SELECT month, year
   FROM Purchase, Show
   WHERE Purchase.showID = Show.id
   GROUP BY year, month
   HAVING SUM(price) > 2000

2. SELECT email
   FROM Purchase
   GROUP BY email
   HAVING COUNT(*) >= 2

3. SELECT itemtype
   FROM Merchandise m, MerchSales ms
   WHERE m.id = ms.merchandiseID
   GROUP BY itemtype
   HAVING SUM(price) > 500

4. SELECT instrument
   FROM Musician
   GROUP BY instrument
   HAVING count(id) > 2

Part 2

1. SELECT eID
   FROM Musician
   WHERE age < (SELECT AVG(age) FROM Musician)

2. SELECT year, month, date, sum(attendanceNumber)
   FROM Show
   GROUP BY year, month, date

3. SELECT instrument, count(*)
   FROM Musician
   WHERE nationality = ‘Canadian”
   GROUP BY Instrument

4. SELECT sp.composer, sp.title
   FROM SongsPerformed sp
   WHERE sp.showID in (SELECT id
   FROM Person p, Purchase pur
   WHERE pur.email = p.email
   GROUP BY pur.showID]
   HAVING AVG(p.age) > (SELECT AVG(p1.age) FROM Person p1))

Part 3

1. UPDATE Merchandise
   SET price = price + 10
   WHERE description LIKE '%book%'

2. UPDATE Merchandise
   SET price = price - 10
   WHERE price > 10 AND description LIKE '%book%'

3. UPDATE Person
   SET age = age - 2
   WHERE name LIKE 'K%'

4. UPDATE Person
   SET age = age + 5
   WHERE name LIKE '_A%'

Part 4

1. SELECT composer, title
   FROM SongsPerformed sp1
   WHERE NOT EXISTS (
   (SELECT eid
   FROM Musician
   WHERE instrument='violin')

   EXCEPT

   (SELECT id
   FROM PerformedIn pi, SongsPerformed sp2
   WHERE pi.showID = sp2.showID AND sp2.composer = sp1.composer AND
   sp2.title = sp1.title))

2.
```
SELECT DISTINCT composer
FROM SongsPerformed sp1
WHERE NOT EXISTS (
        (SELECT email
        FROM Person WHERE age < 30)

        EXCEPT

        (SELECT email
        FROM Purchase, SongsPerformed sp2
        WHERE Purchase.ShowID = sp2.ShowID AND sp2.composer = sp1.composer))
```

3.
```
SELECT ms1.merchandiseID
FROM MerchSales ms1
WHERE NOT EXISTS (
        (SELECT email
        FROM Person WHERE age < 30)

        EXCEPT

        (SELECT DISTINCT email
        FROM MerchSales ms2
        WHERE ms1.merchandiseID = ms2.merchandiseID) );
```

4.
```
SELECT distinct p1.email
FROM Purchase p1
WHERE NOT EXISTS (
        (SELECT DISTINCT showID
        FROM SongsPerformed
        WHERE composer = 'Tchaikovsky')

        EXCEPT

        (SELECT DISTINCT showID
        FROM Purchase p2
        WHERE p1.email = p2.email))
```

5.
```
SELECT eID
FROM Musician m1
WHERE NOT EXISTS ( (SELECT ID
                    FROM Show
                    WHERE year = 2020 AND month = 'January')

                    EXCEPT

                   (SELECT showID
                   FROM PerformedIn pi
                    WHERE m1.eID = pi.id))
```

Part 5

1.
```
CREATE VIEW ShowNumbers(year, month, lowestAttendance) AS
  SELECT year, month, MIN(attendanceNumber) AS lowestAttendance
  FROM Show
  GROUP BY year, month

CREATE VIEW LowestAttendedShowJan2020(showID) AS
  SELECT ID
  FROM Show
  WHERE year = 2020 AND month = 'January' AND attendanceNumber =
        (SELECT lowestAttendance
        FROM ShowNumbers
        WHERE year = 2020 AND month = 'January')

CREATE VIEW LowestAttendedShowFeb2020(showID) AS
  SELECT ID
  FROM Show
  WHERE year = 2020 AND month = 'February' AND attendanceNumber =
         (SELECT lowestAttendance
         FROM ShowNumbers
         WHERE year = 2020 AND month = 'February')

SELECT DISTINCT id
FROM PerformedIn
WHERE ShowID IN (SELECT ShowID FROM LowestAttendedShowJan2020)

INTERSECT

SELECT DISTINCT id
FROM PerformedIn
WHERE ShowID IN (SELECT ShowID FROM LowestAttendedShowFeb2020)
```

2.
```
CREATE VIEW ShowNumbers(year, month, lowestAttendance) AS
  SELECT year, month, MIN(attendanceNumber) AS lowestAttendance
  FROM Show
  GROUP BY year, month

CREATE VIEW LowestAttendedShowJan2020(showID) AS
  SELECT ID
  FROM Show
  WHERE year = 2020 AND month = 'January' AND attendanceNumber =
      (SELECT lowestAttendance
      FROM ShowNumbers
      WHERE year = 2020 AND month = 'January')

CREATE VIEW LowestAttendedShowFeb2020(showID) AS
  SELECT ID
  FROM Show
  WHERE year = 2020 AND month = 'February' AND attendanceNumber =
        (SELECT lowestAttendance
        FROM ShowNumbers
        WHERE year = 2020 AND month = 'February')

SELECT DISTINCT merchandiseID
FROM MerchSales
WHERE ShowID IN (SELECT * FROM LowestAttendedShowJan2020
          UNION
          SELECT * FROM LowestAttendedShowFeb2020)
```