GROUP REPORT

Homework #1: PostgreSQL – SQL
COMP 453

Group Members:

Member 1: Bernard Ofori Boateng

- Scheduled group meetings.
- Group report writing.
- Contributed in DDL/DML script for the Music Database.
- Contributed to developing SQL solutions for all 6 queries.

Member 2: Sung Yu Yeh

- Participated in group meetings to review and discuss each part of queries
- Report writing and double checking the content
- Built DDL/DML script for the Music Database
- Searched the right usages of each query to find solutions for all queries.

Question 1: DDL/DML script for the Music Database using the music1.txt

```
-- Create the recordingLabel table
CREATE TABLE recordingLabel (
    labelID VARCHAR(50) PRIMARY KEY,
    labelName VARCHAR(50),
    location VARCHAR(50)
);

-- Insert data into recordingLabel table
INSERT INTO recordingLabel (labelID, labelName, location)
VALUES
    ('L1', 'A&M Records', 'Los Angeles'),
    ('L2', 'Reprise Records', 'New York'),
    ('L3', 'Disney Records', 'Los Angeles'),
    ('L4', 'Country Club', 'Detroit'),
    ('L5', 'Gray Dot Records', 'Detroit'),
    ('L6', 'Disney Records', 'Detroit'),
    ('L7', 'BMG Records', 'Phoenix'),
    ('L8', 'Universal Records', 'Texas'),
    ('L9', 'Parkwood Entertainment', 'Los Angeles'),
    ('L10', 'Big Machine Records', 'Nashville');

-- Create the artist table
CREATE TABLE artist (
    artistID VARCHAR(50) PRIMARY KEY,
```

```
    firstName VARCHAR(50),
    lastName VARCHAR(50),
    yearBorn NUMERIC
);

-- Insert data into artist table
INSERT INTO artist (artistID, firstName, lastName, yearBorn)
VALUES
    ('A1', 'Bjorn', 'Friedman', 1945),
    ('A2', 'Heidi', 'Helmut', 1945),
    ('A3', 'John', 'Stark', 1971),
    ('A4', 'Michael', 'Agnelo', 1975),
    ('A5', 'Michelle', 'Agnelo', 1975),
    ('A6', 'Francis', 'McDermott', 1960),
    ('A7', 'Steve', 'Nash', 1965),
    ('A8', 'Lisa', 'Raymond', 1973),
    ('A9', 'Janet', 'Brown', 1972),
    ('A10', 'Henry', 'Brown', 1966),
    ('A11', 'Keith', 'Urban', 1970),
    ('A12', 'John', 'Hopkins', 1960),
    ('A13', 'Jim', 'Kate', 1970),
    ('A14', 'Jacky', 'Chen', 1945),
    ('A15', 'Beyonce', 'Carter', 1981),
    ('A16', 'Taylor', 'Swift', 1989);

-- Create the musicalGroup table
CREATE TABLE musicalGroup (
    groupCode VARCHAR(50) PRIMARY KEY,
    groupName VARCHAR(50)
);

-- Insert data into musicalGroup table
INSERT INTO musicalGroup (groupCode, groupName)
VALUES
    ('G1', 'Deutsch Moss'),
    ('G2', 'Sweet Symphony'),
    ('G3', 'Thundering Typhoons'),
    ('G4', 'Great Snakes'),
    ('G5', 'Keith Urban'),
    ('G6', 'Beyond'),
    ('G7', 'Backstreet Boys'),
    ('G8', 'Tomorrow'),
    ('G9', 'Destiny"s child');

-- Create the member table
CREATE TABLE member (
```

```
    groupCode VARCHAR(50),
    artistID VARCHAR(50),
    fromDate NUMERIC,
    toDate NUMERIC,
    PRIMARY KEY (groupCode, artistID, fromDate)
);

-- Insert data into member table
INSERT INTO member (groupCode, artistID, fromDate, toDate)
VALUES
    ('G1', 'A1', 1970, 1990),
    ('G2', 'A1', 1991, 0),
    ('G1', 'A2', 1970, 1990),
    ('G1', 'A3', 1971, 0),
    ('G2', 'A4', 1995, 0),
    ('G5', 'A4', 1997, 0),
    ('G2', 'A5', 1995, 2003),
    ('G3', 'A6', 1985, 0),
    ('G3', 'A7', 2000, 0),
    ('G1', 'A8', 1991, 2003),
    ('G4', 'A9', 1990, 2003),
    ('G1', 'A9', 1980, 2003),
    ('G2', 'A9', 1998, 0),
    ('G4', 'A10', 1990, 0),
    ('G6', 'A10', 1998, 0),
    ('G5', 'A11', 1980, 2003),
    ('G8', 'A10', 1950, 1978),
    ('G7', 'A1', 1980, 0),
    ('G7', 'A2', 1967, 1969),
    ('G7', 'A3', 1980, 1981),
    ('G2', 'A14', 1960, 0),
    ('G9', 'A15', 1990, 0);

-- Create the cd table
CREATE TABLE cd (
    cdCode VARCHAR(50) PRIMARY KEY,
    cdTitle VARCHAR(50),
    numberSold NUMERIC,
    year NUMERIC,
    labelID VARCHAR(50),
    groupCode VARCHAR(50)
);

-- Insert data into cd table
INSERT INTO cd (cdCode, cdTitle, numberSold, year, labelID, groupCode)
VALUES
```

```
    ('C1', 'Goodnight', 500000, 2001, 'L1', 'G1'),
    ('C2', 'Sweet Dreams', 100000, 1990, 'L6', 'G2'),
    ('C3', 'Forgotten', 800000, 1998, 'L4', 'G2'),
    ('C4', 'Friday Night', 800000, 2003, 'L1', 'G3'),
    ('C5', 'Hot As Hell', 750000, 2003, 'L3', 'G3'),
    ('C6', 'Golden Hits', 400000, 1998, 'L3', 'G4'),
    ('C7', 'Broken', 600000, 2003, 'L1', 'G4'),
    ('C8', 'Golden Road', 800000, 2003, 'L3', 'G4'),
    ('C9', 'Function', 100000, 2002, 'L2', 'G4'),
    ('C10', 'Be Here', 790000, 2003, 'L1', 'G5'),
    ('C11', 'Ranch', 450000, 2003, 'L5', 'G2'),
    ('C12', 'Horse of A Different Color', 900000, 2001, 'L1', 'G5'),
    ('C13', 'Rain', 450000, 2000, 'L1', 'G5'),
    ('C14', 'Color', 40000, 2000, 'L4', 'G1'),
    ('C15', 'Wonderful', 89000, 1997, 'L5', 'G5'),
    ('C16', 'Happy', 99000, 1994, 'L6', 'G5'),
    ('C17', 'Moon Lake', 25000, 1998, 'L4', 'G6'),
    ('C18', 'Summer Fiesta', 75000, 2003, 'L5', 'G6'),
    ('C19', 'Kettie', 800000, 2001, 'L6', 'G6'),
    ('C20', 'Celebrate', 790000, 2003, 'L2', 'G6'),
    ('C21', 'Sunshine', 34000, 2001, 'L8', 'G6'),
    ('C22', 'Lemonade', 500000, 2016, 'L9', 'G9');

-- Create the topCDs table
CREATE TABLE topCDs (
    cdCode VARCHAR(50),
    year NUMERIC,
    rating NUMERIC,
        PRIMARY KEY (cdCode, year)
);

-- Insert data into topCDs table
INSERT INTO topCDs (cdCode, year, rating)
VALUES
    ('C1', 2001, 2),
    ('C2', 1990, 1),
    ('C3', 1998, 6),
    ('C4', 2003, 3),
    ('C5', 2003, 9),
    ('C6', 1998, 1),
    ('C7', 2003, 2),
    ('C8', 2003, 1),
    ('C9', 2002, 5),
    ('C10', 2003, 4),
    ('C11', 2003, 11),
    ('C12', 2001, 1),
```

```
    ('C13', 2000, 15),
    ('C22', 2016, 3);

-- Create the song table
CREATE TABLE song (
    songCode VARCHAR(50) PRIMARY KEY,
    songTitle VARCHAR(50)
);

-- Insert data into song table
INSERT INTO song (songCode, songTitle)
VALUES
    ('S1', 'Sweet Dreams'),
    ('S2', 'Goodnight'),
    ('S3', 'My Lullaby'),
    ('S4', 'I Have A Dream'),
    ('S5', 'Forgotten'),
    ('S6', 'How Could You'),
    ('S7', 'Breathless'),
    ('S8', 'My Oh My'),
    ('S9', 'Ooh La La'),
    ('S10', 'Fire'),
    ('S11', 'Hot As Hell'),
    ('S12', 'Goldilocks'),
    ('S13', 'One Of Us'),
    ('S14', 'Mamma Mia'),
    ('S15', 'Broken'),
    ('S16', 'Life Or Death'),
    ('S17', 'Days Go By'),
    ('S18', 'Another Day in Paradise'),
    ('S19', 'Smile'),
    ('S20', 'Formation'),
    ('S21', 'Freedom'),
    ('S22', 'Love Drought'),
    ('S23', 'Lover'),
    ('S24', 'Anti-Hero'),
    ('S25', 'All too well');

-- Create the composedOf table
CREATE TABLE composedOf (
    cdCode VARCHAR(50),
    songCode VARCHAR(50),
    trackNumber NUMERIC,
    PRIMARY KEY (cdCode, songCode)
);
```

-- Insert data into composedOf table

```
INSERT INTO composedOf (cdCode, songCode, trackNumber)
VALUES
    ('C1', 'S1', 1),
    ('C1', 'S2', 2),
    ('C1', 'S3', 3),
    ('C1', 'S12', 4),
    ('C1', 'S11', 5),
    ('C1', 'S13', 6),
    ('C1', 'S10', 7),
    ('C1', 'S8', 8),
    ('C2', 'S4', 1),
    ('C3', 'S5', 1),
    ('C3', 'S6', 2),
    ('C4', 'S7', 1),
    ('C4', 'S8', 2),
    ('C4', 'S9', 3),
    ('C5', 'S10', 1),
    ('C5', 'S11', 2),
    ('C5', 'S8', 3),
    ('C6', 'S12', 1),
    ('C6', 'S13', 2),
    ('C6', 'S14', 3),
    ('C7', 'S15', 1),
    ('C7', 'S16', 2),
    ('C7', 'S10', 3),
    ('C8', 'S17', 1),
    ('C8', 'S18', 2),
    ('C8', 'S1', 3),
    ('C8', 'S2', 4),
    ('C8', 'S7', 5),
    ('C8', 'S19', 6),
    ('C8', 'S8', 7),
    ('C8', 'S9', 8),
    ('C8', 'S11', 9),
    ('C8', 'S12', 10),
    ('C9', 'S18', 1),
    ('C9', 'S1', 2),
    ('C9', 'S2', 3),
    ('C9', 'S3', 4),
    ('C9', 'S4', 5),
    ('C9', 'S5', 6),
    ('C9', 'S6', 7),
    ('C9', 'S7', 8),
    ('C9', 'S8', 9),
    ('C10', 'S1', 1),
```

```
    ('C10', 'S2', 2),
    ('C11', 'S2', 1),
    ('C11', 'S3', 2),
    ('C12', 'S3', 1),
    ('C13', 'S1', 1),
    ('C13', 'S7', 2),
    ('C13', 'S6', 3),
    ('C13', 'S2', 4),
    ('C13', 'S3', 5),
    ('C13', 'S4', 6),
    ('C13', 'S5', 7),
    ('C14', 'S5', 1),
    ('C14', 'S6', 2),
    ('C14', 'S7', 3),
    ('C14', 'S8', 4),
    ('C14', 'S9', 5),
    ('C15', 'S6', 1),
    ('C16', 'S8', 1),
    ('C16', 'S9', 2),
    ('C17', 'S9', 1),
    ('C18', 'S1', 1),
    ('C18', 'S10', 2),
    ('C18', 'S13', 3),
    ('C19', 'S1', 1),
    ('C19', 'S2', 2),
    ('C19', 'S3', 3),
    ('C20', 'S4', 1),
    ('C20', 'S14', 2),
    ('C20', 'S10', 3),
    ('C21', 'S14', 1),
    ('C21', 'S7', 2),
    ('C21', 'S18', 3),
    ('C21', 'S13', 4),
    ('C21', 'S1', 5),
    ('C21', 'S19', 6),
    ('C21', 'S17', 7),
    ('C21', 'S5', 8),
    ('C22', 'S20', 1),
    ('C22', 'S21', 2),
    ('C22', 'S22', 3);
-- Create the writtenBy table
CREATE TABLE writtenBy (
    songCode VARCHAR(50),
    artistID VARCHAR(50),
    PRIMARY KEY (songCode, artistID)
);
```

```
-- Insert data into writtenBy table
INSERT INTO writtenBy (songCode, artistID)
VALUES
    ('S1', 'A1'),
    ('S2', 'A1'),
    ('S3', 'A12'),
    ('S4', 'A3'),
    ('S5', 'A1'),
    ('S5', 'A2'),
    ('S6', 'A2'),
    ('S7', 'A6'),
    ('S8', 'A7'),
    ('S9', 'A7'),
    ('S10', 'A8'),
    ('S11', 'A3'),
    ('S12', 'A8'),
    ('S13', 'A8'),
    ('S14', 'A8'),
    ('S15', 'A8'),
    ('S16', 'A9'),
    ('S17', 'A4'),
    ('S18', 'A10'),
    ('S19', 'A13'),
    ('S20', 'A15'),
    ('S21', 'A15'),
    ('S22', 'A15'),
    ('S23', 'A16'),
    ('S24', 'A16'),
    ('S25', 'A16');

-- Create the topSongs table
CREATE TABLE topSongs (
    songCode VARCHAR(50),
    year NUMERIC,
    rating NUMERIC,
          PRIMARY KEY (songCode,year)
);

-- Insert data into topSongs table
INSERT INTO topSongs (songCode, year, rating)
VALUES
    ('S1', 1987, 5),
    ('S2', 1987, 1),
    ('S4', 1990, 1),
    ('S5', 1998, 3),
```

('S6', 1998, 5),
('S8', 2000, 4),
('S10', 1993, 2),
('S11', 1993, 13),
('S12', 1998, 1),
('S14', 1998, 4),
('S15', 1999, 1),
('S16', 1999, 3),
('S17', 2003, 1),
('S18', 2003, 12),
('S19', 2000, 1),
('S20', 2016, 1),
('S23', 2019, 2),
('S24', 2022, 1),
('S25', 2012, 4);

Question 2: SQL Queries and Solutions

Query 1 : List the artists that belong to a musical group, born no later than 1985. Order the list by groupName in ascending order.
(groupName, firstName, lastName, yearBorn)

SQL Query:

```
SELECT MG.groupName, A.firstName, A.lastName, A.yearBorn
FROM musicalGroup MG, member M, artist A
WHERE MG.groupCode = M.groupCode
AND M.artistID = A.artistID
AND A.yearBorn <= 1985
ORDER BY MG.groupName ASC;
```

Output:

groupName	firstName	lastName	yearBorn
Backstreet Boys	Heidi	Helmut	1945
Backstreet Boys	Bjorn	Friedman	1945
Backstreet Boys	John	Stark	1971
Beyond	Henry	Brown	1966
Destiny's child	Beyonce	Carter	1981
Deutsch Moss	Janet	Brown	1972
Deutsch Moss	Heidi	Helmut	1945
Deutsch Moss	John	Stark	1971
Deutsch Moss	Lisa	Raymond	1973
Deutsch Moss	Bjorn	Friedman	1945
Great Snakes	Janet	Brown	1972
Great Snakes	Henry	Brown	1966
Keith Urban	Keith	Urban	1970
Keith Urban	Michael	Agnelo	1975
Sweet Symphony	Jacky	Chen	1945
Sweet Symphony	Bjorn	Friedman	1945
Sweet Symphony	Janet	Brown	1972
Sweet Symphony	Michelle	Agnelo	1975
Sweet Symphony	Michael	Agnelo	1975
Thundering Typhoons	Francis	McDermott	1960
Thundering Typhoons	Steve	Nash	1965
Tomorrow	Henry	Brown	1966

Question 2 : For every musical group, find the number of CDs rated in the top 10. List the count of CDs in descending order. You don't need to include the artists with zero top-10 CDs. (groupName, countOfTopCDs)

SQL Query:

```
SELECT MG.groupName, COUNT(*) AS countOfTopCDs
FROM musicalGroup MG, cd C, topCDs TC
WHERE MG.groupCode = C.groupCode
AND C.cdCode = TC.cdCode
GROUP BY MG.groupName
HAVING COUNT(*) > 0
ORDER BY countOfTopCDs DESC;
```

Output:

groupName	countOfTopCDs
Great Snakes	4
Keith Urban	3
Sweet Symphony	3
Thundering Typhoons	2
Destiny's child	1
Deutsch Moss	1

Question 3 : List the maximum, minimum and average number of CDs sold per record label. Order by the labelName in ascending order.
(labelName, maxNumber, minNumber, avgNumber)

SQL Query:

```
SELECT RL.labelName, MAX(C.numberSold) AS maxNumber, MIN(C.numberSold) AS
minNumber, AVG(C.numberSold) AS avgNumber
FROM recordinglabel RL, cd C
WHERE C.labelid = RL.labelid
GROUP BY RL.labelName
ORDER BY labelName ASC;
```

Output:

labelName	maxNumber	minNumber	avgNumber
A&M Records	900000	450000	673333.333333333000
Country Club	800000	25000	288333.333333333000
Disney Records	800000	99000	491500.000000000000
Gray Dot Records	450000	75000	204666.666666667000
Parkwood Entertainment	500000	500000	500000.000000000000
Reprise Records	790000	100000	445000.000000000000
Universal Records	34000	34000	34000.000000000000

Question 4 : How many top songs has each recording label produced? Order by the number of songs in descending order. Note: In this database, a song can be part of multiple CDs. Hint: Use Distinct to get the most accurate result.
(labelName, numberOfSong)

SQL Query:

```
SELECT RL.labelName, COUNT(DISTINCT S.songCode) AS numberOfSong
FROM recordinglabel RL, cd C, composedOf CO, song S
WHERE RL.labelID = C.labelID
AND C.cdCode = CO.cdCode
AND CO.songCode = S.songCode
AND S.songCode IN (SELECT TS.songCode FROM topSongs TS)
GROUP BY RL.labelName
ORDER BY numberOfSong DESC;
```

Output:

labelName	numberOfSong
A&M Records	11
Disney Records	11
Reprise Records	9
Universal Records	6
Gray Dot Records	4
Country Club	3
Parkwood Entertainment	1

Question 5 : Identify the artists that don't have a top 3 song and were born no earlier than 1970. Sort the output by artist's last name.
(firstName, lastName, yearBorn)

SQL Query:

```
SELECT A.firstName, A.lastName, A.yearBorn
FROM artist A
WHERE A.yearBorn >= 1970
AND A.artistID NOT IN (
   SELECT DISTINCT WB.artistID
   FROM writtenBy WB
   WHERE WB.songCode IN (
      SELECT TS.songCode
      FROM topSongs TS
      WHERE TS.rating <= 3
   )
)
ORDER BY A.lastName;
```

Output:

firstName	lastName	yearBorn
Michelle	Agnelo	1975
Keith	Urban	1970

Question 6 : Identify the artists that have at least 2 top songs (wrote 2+ —top songs) and at least 2 top cd (wrote 2+ song(s) for a CD that is a top --CD). Sort the output by artist ID. (artistID, firstName, lastName, countOfTopSongs, countOfTopCDs)

SQL Query:

```
SELECT T1.artistID, T1.firstName, T1.lastName, T1.countOfTopSongs, T2.countOfTopCDs
FROM (
   SELECT A.artistID, A.firstName, A.lastName, COUNT(*) AS countOfTopSongs
   FROM artist A
   JOIN writtenBy WB ON A.artistID = WB.artistID
   JOIN topSongs TS ON WB.songCode = TS.songCode
   GROUP BY A.artistID, A.firstName, A.lastName
   HAVING COUNT(*) >= 2
) T1,
(
   SELECT A.artistID, COUNT(DISTINCT TC.cdCode) AS countOfTopCDs
   FROM artist A
   JOIN writtenBy WB ON A.artistID = WB.artistID
   JOIN composedOf CO ON WB.songCode = CO.songCode
   JOIN topCDs TC ON CO.cdCode = TC.cdCode
   GROUP BY A.artistID
   HAVING COUNT(DISTINCT TC.cdCode) >= 2
) T2
WHERE T1.artistID = T2.artistID
ORDER BY T1.artistID ASC;
```

Output:

artistID	firstName	lastName	countOfTopSongs	countOfTopCDs
A1	Bjorn	Friedman	3	7
A2	Heidi	Helmut	2	3
A3	John	Stark	2	6
A8	Lisa	Raymond	4	5