**SQL FINAL PROJECT - Andrea Amel -**

Overview

The final four class hours of this course will be the final project. The project will use all the SQL skills we have covered so far. Be sure to record all code you write in the project workbook.

**Part 1**

**Task 1**

**Select b.bandname, a.releasedate from band b**

**join album a on b.idband = a.idband**

**order by releasedate desc;**

**Task 2**

Drop Table Records also sends bands to play at different venues or events.

Some venues request bands that feature certain instruments. Write a query that shows all of the players that utilize drums along with the bands that they are a part of. You should only have one column that shows the full player name (i.e., the player’s first and last name should not be split into two columns).

Copy the text of your query below.

**select concat(pfname, " ", plname) as player\_name, i.instrument**

**from band\_db.player as p**

**join instrument i on i.InstID = p.InstID;**

**Task 3**

It is important that Drop Table Records has a diversity of different bands signed to its label. We want many artists who represent different styles of music.

One way we determine the diversity in music is by looking at how many bands feature a certain instrument. Write a query that describes the number of instruments used by each band. (Hint: some bands may have multiple players using the same instrument.)

Copy the text of your query below.

**select distinct count(i.instrument), p.idplayer, b.bandname from player p**

**join instrument i on p.InstID = i.InstID**

**join band b on p.idband = b.idband group by bandname;**

**Task 4**

**Select count(p.InstID), i.instrument from player p**

**join instrument i on p.InstID = i.InstID**

**group by instrument order by instrument desc;**

**Task 5**

**select idalbum, albumname, releasedate from band\_db.album**

**where releasedate is null or albumname is null**

--These null values should be cross referenced with another table that has a common foreign key to find missing information. Otherwise, it might make sense to collaborate with a team member who has the information or search for the information and then update the table to insert the values into the table. It is necessary to document that these values are null in a text for future updates.

**Part 2**

**Task 1**

We need to add more bands to the band table. Here’s a list of the bands we’d like to add:

Band name

Weezer

TLC

Paramore

Blackpink

Vampire Weekend

Hint: When adding values to a table, take a look at the table schema to see if a primary key needs to be identified or if it will be auto-generated.

Copy the text of your query below.

**Insert into band(aid, bandname)**

**values (1, 'Weezer')**

**Insert into band(aid, bandname)**

**values (1, 'TLC')**

**Insert into band(aid, bandname)**

**values (1, 'Paramore')**

**Insert into band(aid, bandname)**

**values (1, 'Blackpink')**

**Insert into band(aid, bandname)**

**values (1, 'Vampire Weekend')**

– – It was not necessary to enter ‘idband’ for the primary key as it was auto-generated. After each entry I put the query Select \* from band to double check for accuracy.

**Task 2**

Now that we have added more bands, we need to ensure that we add the band members to the appropriate table. Which table would we use to add the names of band members?

Enter your answer below.

**We would add the names of the band members to the ‘player’ table, since it has fields for ‘pfname’ and ‘plname’.**

**Task 3**

Using the table you identified in Task 2, add the following values

Copy the text of your query below.

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (3, 22, 'Rivers', 'Cuomo', 'Rochester', 'New York');**

**Update player Set homestate = 'NY' where idplayer = 30**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (1, 22, 'Brian', 'Bell', 'Iowa City', 'Iowa');**

**Update player Set homestate = 'IA' where idplayer = 31**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (4, 22, 'Patrick', 'Wilson', 'Buffalo', 'New York');**

**Update player Set homestate = 'NY' where idplayer = 32**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (2, 22, 'Scott', 'Shriner', 'Toledo', 'Ohio');**

**Update player Set homestate = 'OH' where idplayer = 33**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (3, 23, 'Tionne', 'Watkins', 'Des Moines', 'Iowa');**

**Update player Set homestate = 'IA' where idplayer = 34**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (3, 23, 'Rozonda', 'Thomas', 'Des Moines', 'Iowa');**

**Update player Set homecity = 'Columbus', homestate = 'GA' where idplayer = 35**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (3, 24, 'Hayley', 'Williams', 'Franklin', 'TN');**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (1, 25, 'Taylor', 'York', 'Nashville', 'TN');**

**Update player set idband = 24 where idplayer = 37**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (4, 24, 'Zac', 'Farro', 'Voorhees Township', 'NJ');**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (3, 25, 'Jisoo', 'Kim', ' ', 'South Korea');**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (3, 25, 'Jennie', 'Kim', ' ', 'South Korea');**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (3, 25, 'Roseanne', 'Park', ' ', 'New Zealand');**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (3, 25, 'Lisa', 'Manoban', ' ', 'South Thailand');**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (3, 26, 'Ezra', 'Koenig', 'New York', 'NY');**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (2, 26, 'Chris', 'Baio', 'Bronxville', 'NY');**

**Insert into player(InstID, idband, pfname, plname, homecity, homestate)**

**Values (4, 26, 'Chris', 'Tomson', 'Upper Freehold Township', 'NJ');**

– – I repeated this query for each entry and entered Select \* from player to double check accuracy. If I found anything I needed to fix, I used this query- Update player Set homestate = 'NY' where idplayer = 30

– – I wanted to make the state name consistent with the rest of the table so I update ‘New York’ to ‘NY’, as I thought a shorter set of characters would be more efficient to process. Because this is a long set of separate queries, I did not post them all here since it repeats the same basic query over and over. I am curious to know if there was a more efficient way to do this that I missed. I would love feedback as to the process if there is a more efficient way!

**Task 4**

Drop Table Records has signed a contract with a new venue! A new venue should be added to the venue table.

**Insert into venue(vname, city, state, zipcode, seats)**

**values('Twin City Rock House', 'Minneapolis', 'MN', 55414, 2000);**

– – At first I typed the full state name for Minnesota, but I got an error stating it was too long for the row, so I changed it to the abbreviation MN and it worked.

**Task 5**

Which state has the largest amount of seating available (regardless of the number of venues at the state)?

Hint: We are trying to determine the total number of seats for each state.

Copy the text of your query below.

**Select state, SUM(seats) from venue group by State;**

**– – California (CA) has the largest number of seats (36,562).**

**Part 3**

**Task 1**

We need to add some data on upcoming performances for some of the artists. Which table should we use to add this information?

Copy the text of your query below.

**– – The ‘gig’ table has columns for upcoming performance information, such as venue, band ID, date, and attendees.**

**Task 2**

Using the table you mentioned in Task 1 (above), add the following information

**Insert into band\_db.gig(idvenue, idband, gigdate, numattendees)**

**values(4, 2, '2022-05-05', 19000)**

**Insert into band\_db.gig(idvenue, idband, gigdate)**

**values(12, 26, '2022-04-15')**

**Insert into band\_db.gig(idvenue, idband, gigdate, numattendees)**

**values(8, 23, '2022-06-07', 18000)**

**Insert into band\_db.gig(idvenue, idband, gigdate, numattendees)**

**values(2, 22, '2022-07-03', 175)**

– – I was having trouble creating a join where I could see all idvenue and idband matched up with their names for reference, so I simply pulled up the tables ‘venue’ and ‘band’ for reference and wrote down the ids along with their names. At first I forgot that the datetime entry needed ‘ ‘ quotes around it because of the dashes, but I fixed it. I removed numattendees from the fields for the second entry because it was a null value.

**Task 3**

Are any of the venues oversold?

Copy the text of your query below.

**Select g.idvenue, g.idband, g.numattendees, v.seats, v.vname from gig g**

**join venue v on g.idvenue = v.idvenue;**

– – The TLC concert at the SAP Center is oversold. The number of attendees is 18,000 but the number of seats is 17, 562.

**Task 4**

We just got word that Vampire Weekend can expect 1,750 guests. Write a query to update the table accordingly.

**Copy the text of your query below.**

**UPDATE gig**

**set numattendees = 1750**

**where GigID = 2;**

– – This gig is not oversold either as the seat capacity is 2000 at the venue.

**Task 5**

We just got an update that the expected number of attendees at the River Club for Weezer will only have 125 guests. Write a query to update the table accordingly.

Copy the text of your query below.

**Update gig**

**Set numattendees = 125**

**where GigID = 4;**

– – I double checked the gig table to make sure the change happened and it had not, so I ran the query again, and the second time it worked when I checked.

**Task 6**

Create a view (called vw\_giginfo) that will show the band, the dates they will play, the venue they will play at, the number of attendees, and the venue capacity. For this view, also create a column that describes what percentage of the venue capacity was utilized.

Copy the text of your query below.

**Create view band\_db.vw\_giginfo AS**

**Select b.bandname, v.vname, g.gigdate, g.numattendees, v.seats, 100\*(g.numattendees/v.seats) AS percentage\_utilized from venue v**

**join gig g on v.idvenue = g.idvenue**

**join band b on g.idband = b.idband;**

– – After I completed this, I double checked that my view worked using this query:

**Select \* from band\_db.vw\_giginfo** (Thank you Dr. Monday!)

**Part 4**

**Task 1**

Create a stored procedure that lists all of the venues that can handle more than 10,000 guests.

Copy the text of your query below.

**Create Procedure Large\_Venues1()**

**BEGIN**

**Select vname, seats from venue where seats >= 10000 order by seats asc;**

**END //**

**DELIMITER ;**

**Call Large\_Venues1()**

– – I first tested the query from the select statement to see that it worked. It did, and I liked the ascending order of the seat numbers. After creating the procedure and correcting a syntax error, I called the procedure and it displayed.

**Task 2**

Create a stored procedure that lists all of the players that come from a specific state. We want to see (once we run this stored procedure), what bands they are a part of, their full name (in one column), and the state they are from.

**Create Procedure Player\_Band\_HomeState5(IN State\_Name varchar(50))**

**BEGIN**

**Select p.pfname, p.plname, p.homestate, b.bandname from band\_db.band b join band\_db.player p on b.idband = p.idband where p.homestate = State\_Name;**

**END //**

**DELIMITER ;**

**Call Player\_Band\_Homestate5('NY');**