

SQL Code Challenge: Emp Table

NOTES:

- I would change the "mgr" column to be a VARCHAR type as there should not be any numeric operations performed on that number.
- The hiredate format change would be set to the default server settings for displaying dates. Since the requested format did not match PostgreSQL 11's default format, it has been left as a string of text. The DATE_PART extraction function does not generate a 2 digit format so it would have to be hard coded in the output string, but since this would most likely be left as a datetime data type in a database changing the default format settings would be a better approach than code manipulation.
- Ran into functionality issues with my version of PostgreSQL in PgAdmin 4 as it did not have some of the newer functions. (e.g. PIVOT, STRING_SPLIT, CROSSTAB)

1. Let's clean up this table, your query should incorporate the following: (Est. Time: 25 min)

- Separate the two "login_id's into different columns.
- Label one column "phone_id" (id with four numbers) and the other "chat_id" (id that starts with "005")
- Change the format of the "ENAME" & "MNAME" column to "First Name and Last Name" (e.g., Peter Parker) and sort "JOB" in ascending order but make sure "President" is at the top of the list.
- Transform the "HIREDATE" column to show only the date, and format it by "DD-MM-YYYY" (e.g. "03-05-2021")
- Create a new column labeled "ETENURE" and calculate the employee's tenure from their hire date to "2021-05-03" and display it by years, months, and days.
- Transform all null values to something other than null, you will need to decide what to display instead of the null values.

1. Import CSV to DB

```
CREATE TABLE emp (  
    empno NUMERIC(10,0) NOT NULL,  
    ename VARCHAR(100) NOT NULL,  
    login_id VARCHAR(100),  
    job VARCHAR(100) NOT NULL,  
    mgr NUMERIC(10,0),  
    mname VARCHAR(100),  
    hiredate DATE NOT NULL,  
    sal NUMERIC(10,0) NOT NULL,
```

```

        commission NUMERIC(10,0),
        team_name VARCHAR(100) NOT NULL
    );



```

2. Create separate table with chat login ids

```

SELECT empno, login_id as chat_id
INTO chat_id_tbl
FROM emp
WHERE login_id LIKE '005%';
SELECT * FROM chat_id_tbl;

```




	 empno numeric (10)	 chat_id character varying (100)
1	74992	00550000000ssw5AAA
2	75210	0055000000176uYAAQ
3	76546	005500000003kp63AAA
4	78449	005500000006rM4ZAAU
5	75665	005500000003ko7jAAA
6	76984	005500000003kp8aAAA
7	77829	005500000003kp8jAAA
8	77882	005500000005JDwqAAG
9	79029	0052J0000008GXsyQAG

3. Create separate table with phone login ids

```

SELECT empno, login_id as phone_id
INTO phone_id_tbl
FROM emp
WHERE LENGTH(login_id) = 4;

```

	 empno numeric (10) 	phone_id character varying (100) 
1	73694	4174
2	78766	5102
3	79003	5166
4	79348	5293
5	74992	4524
6	75210	4540
7	76546	4814
8	78449	5047
9	75665	4789
10	76984	4817
11	77829	4860
12	77882	4896
13	79029	5229
14	78393	4970

4. Change the format of the “ENAME” & “MNAME” column to “First Name and Last Name” (e.g., Peter Parker)

-- Create table with correctly formatted employee names

SELECT DISTINCT (empno),

-- PostgreSQL does not use String_Split function so used their version which is SPLIT_PART

SPLIT_PART(ename,',',2)|| ' ' || SPLIT_PART(ename,',',1) AS ename

INTO ename_split

FROM emp

WHERE ename IS NOT NULL;

	empno numeric (10)	ename text
1	76984	Bruce Wayne
2	78393	Ted Mosbey
3	73694	Barney Stinson
4	77829	Tony Stark
5	79029	Hope Van Dyne
6	79003	Jason Bourne
7	78449	Steve Rogers
8	75665	Alexander Luthor
9	75210	Clark Kent
10	78766	Ethan Hunt
11	76546	Pete Ross
12	74992	Lana Lang
13	77882	Chloe Sullivan
14	79348	Bucky Barnes

-- Create table with manager name column alteration from "LastName, FirstName" to "FirstName LastName"

```
SELECT DISTINCT (mgr),
    SPLIT_PART(mname,',',2)|| ' ' || SPLIT_PART(mname,',',1) AS mname
INTO mname_split
FROM emp
WHERE mname IS NOT NULL;
```

	mgr numeric (10)	mname text
1	75665	Alexander Luthor
2	76984	Bruce Wayne
3	78393	Ted Mosbey
4	77829	Tony Stark

5. Transform the "HIREDATE" column to show only the date, and format it by "DD-MM-YYYY" (e.g. "03-05-2021")

6. Create a new column labeled "ETENURE" and calculate the employee's tenure from their hire date to "2021-05-03" and display it by years, months, and days.

-- Date conversion table for calculating tenure and "hiredate" format manipulation

```

SELECT DISTINCT (empno),
    -- PostgreSQL does not use DateDiff function so used their version which is AGE and
    -- has a default output
    AGE('2021-05-03', hiredate) as etenure,
    -- PostgreSQL 11 has preset date format that is difficult to change. Work around by
    -- converting to string format. DATE_PART function works similar to EXTRACT() function in
    -- PostgreSQL 11.
    DATE_PART('day', hiredate) || '-' || date_part('month', hiredate) || '-' || date_part('year',
    hiredate) as hiredate
INTO emp_tenure
FROM emp;

```

7. Combine generated tables into new clean table “emp_clean”

```

SELECT DISTINCT (e.empno),
    es.ename,
    e.job,
    e.mgr,
    ms.mname,
    et.hiredate,
    et.etenure,
    e.sal,
    e.commission,
    e.team_name,
    cht.chat_id,
    phn.phone_id

INTO emp_clean
FROM emp as e
-- join chat_id_tbl to add chat_id column
LEFT JOIN chat_id_tbl as cht
ON e.empno = cht.empno
-- join phone_id_tbl to add phone_id column
LEFT JOIN phone_id_tbl as phn
ON e.empno = phn.empno
-- join ename_split to add fixed ename column
LEFT JOIN ename_split as es
ON e.empno = es.empno
-- join mname_split to add fixed mname column
LEFT JOIN mname_split as ms
ON e.mgr = ms.mgr
-- join emp_tenure
LEFT JOIN emp_tenure as et

```

ON e.empno = et.empno;

8. Transform all null values to something other than null, you will need to decide what to display instead of the null values.

-- Populate null values with appropriate filler.

-- Numerical values are changed from "null" to "0"

UPDATE emp_clean

SET mgr = 0

WHERE

mgr IS NULL;

UPDATE emp_clean

SET commission = 0

WHERE

commission IS NULL;

-- string values are changed from "null" to "none."

UPDATE emp_clean

SET mname = 'none'

WHERE

mname IS NULL;

UPDATE emp_clean

SET chat_id = 'none'

WHERE

chat_id IS NULL;

-- Change the data types for clarity

ALTER TABLE emp_clean

ALTER COLUMN ename TYPE VARCHAR(100),

ALTER COLUMN mname TYPE VARCHAR(100);

	empno numeric (10)	ename character varying (100)	job character varying (100)	mgr numeric (10)	mname character varying (100)	hiredate text	etenure interval	sal numeric (10)	commission numeric (10)	team_name character varying (100)	chat_id character varying (100)	phone_id character varying (100)
1	74992	Lana Lang	SalesPerson	76984	Bruce Wayne	30-7-2001	19 years 9 ..	80526	16994	Team C	005500000000aw5AAA	4524
2	75210	Clark Kent	SalesPerson	76984	Bruce Wayne	1-7-2003	17 years 10..	62911	28323	Team C	0055000000176uYAAQ	4540
3	76546	Pete Ross	SalesPerson	76984	Bruce Wayne	16-6-2020	10 mons 1..	62911	23304	Team C	005500000003kp6SAAA	4814
4	75665	Alexander Luthor	Manager	78393	Ted Mosbey	1-7-2003	17 years 10..	149727	0	Team B	005500000003ko7JAAA	4789
5	76984	Bruce Wayne	Manager	78393	Ted Mosbey	16-6-2020	10 mons 1..	143436	0	Team C	005500000003kp8aAAA	4817
6	77829	Tony Stark	Manager	78393	Ted Mosbey	21-9-2003	17 years 7 ..	123305	0	Team A	005500000003kp8JAAA	4860
7	77882	Chloe Sullivan	Analyst	78393	Ted Mosbey	31-5-2004	16 years 11..	150986	0	Team B	005500000005J0wqAAG	4896
8	78449	Steve Rogers	SalesPerson	77829	Tony Stark	12-8-2010	10 years 8 ..	75493	0	Team C	005500000006rM4ZAAU	5047
9	79029	Hope Van Dyne	Analyst	78393	Ted Mosbey	8-1-2006	15 years 3 ..	150986	0	Team B	0052J0000008G6XyQAG	5229
10	73694	Blamey Stinson	Clerk	75665	Alexander Luthor	30-7-2001	19 years 9 ..	40263	0	Team B	none	4174
11	78766	Ethan Hunt	Clerk	75665	Alexander Luthor	7-11-2005	15 years 5 ..	55361	0	Team B	none	5102
12	79003	Jason Bourne	Clerk	75665	Alexander Luthor	7-11-2005	15 years 5 ..	47812	0	Team C	none	5166
13	79348	Bucky Barnes	Clerk	75665	Alexander Luthor	8-1-2006	15 years 3 ..	65427	0	Team A	none	5293
14	78393	Ted Mosbey	President	0	none	12-8-2010	10 years 8 ..	251643	0	Team A	none	4970

9. Sort by “JOB” in ascending order but make sure “President” is at the top of the list.

```

SELECT *
INTO emp_clean_sorted
FROM emp_clean
ORDER BY
  -- identifies "President" in job column and places that row first, then sorts the remaining in
  ascending order
  CASE
    WHEN job = 'President' then 0
  else 1
end,
job ASC;

```

	empno numeric (10)	ename character varying (100)	job character varying (100)	mgr numeric (10)	mname character varying (100)	hiredate text	etenure interval	sal numeric (10)	commission numeric (10)	team_name character varying (100)	chat_id character varying (100)	phone_id character varying (100)
1	78393	Ted Mosbey	President	0	none	12-8-2010	10 years 8 mons 22 days	251643		0 Team A	none	4970
2	79029	Hope Van Dyne	Analyst	78393	Ted Mosbey	8-1-2006	15 years 3 mons 26 days	150986	0	Team B	0052J000008GXyQAG	5229
3	77882	Chloe Sullivan	Analyst	78393	Ted Mosbey	31-5-2004	16 years 11 mons 3 days	150986	0	Team B	00550000005JdwqAAG	4896
4	78766	Ethan Hunt	Clerk	75665	Alexander Luthor	7-11-2005	15 years 5 mons 26 days	55361	0	Team B	none	5102
5	79003	Jason Bourne	Clerk	75665	Alexander Luthor	7-11-2005	15 years 5 mons 26 days	47812	0	Team C	none	5166
6	79348	Bucky Barnes	Clerk	75665	Alexander Luthor	8-1-2006	15 years 3 mons 26 days	65427	0	Team A	none	5293
7	73694	Barney Stinson	Clerk	75665	Alexander Luthor	30-7-2001	19 years 9 mons 4 days	40263	0	Team B	none	4174
8	75665	Alexander Luthor	Manager	78393	Ted Mosbey	1-7-2003	17 years 10 mons 2 days	149727	0	Team B	00550000003ko7JAAA	4789
9	76984	Bruce Wayne	Manager	78393	Ted Mosbey	16-6-2020	10 mons 17 days	143436	0	Team C	00550000003kp8aAAA	4817
10	77829	Tony Stark	Manager	78393	Ted Mosbey	21-9-2003	17 years 7 mons 12 days	123305	0	Team A	00550000003kp8aAAA	4860
11	78449	Steve Rogers	SalesPerson	77829	Tony Stark	12-8-2010	10 years 8 mons 22 days	75493	0	Team C	00550000006nMAZAAU	5047
12	75210	Clark Kent	SalesPerson	76984	Bruce Wayne	1-7-2003	17 years 10 mons 2 days	62911	28323	Team C	0055000000176uYAAQ	4540
13	76546	Pete Ross	SalesPerson	76984	Bruce Wayne	16-6-2020	10 mons 17 days	62911	23304	Team C	00550000003kp63AAA	4814
14	74992	Lana Lang	SalesPerson	76984	Bruce Wayne	30-7-2001	19 years 9 mons 4 days	80526	16994	Team C	005500000000sswSAAA	4524

2. Let's assume that the CSV file we started with is a live source where our "EMP" table data comes from, you successfully created a new table from that source but there's one problem, the table isn't automatically updated, which means we won't always have the latest employee data. Not to worry I have a solution for you. (Est. Time 40 Min)

- Your task is to create a stored procedure that we can use to create a nightly job to automate the process of loading new data into our "EMP" table.
- To avoid duplicating the existing data you'll need to insert data into our target table (new table you just created) from the results of the source table.
- In other words, synchronize the two tables (Source and Target Tables) by inserting, updating, or deleting rows in one table based on differences found in the other table.

-- Create procedure with input parameter for exported source table and to-be-updated target_table

```
CREATE PROCEDURE update_target_emp(source_table, target_table)
```

```
AS
```

```
BEGIN
```

```
    -- create chat_id table
```

```
    SELECT empno, login_id as chat_id
```

```
    INTO chat_id_tbl
```

```
    FROM source_table
```

```
    WHERE login_id LIKE '005%';
```

```
    -- create phone_id table
```

```
    SELECT empno, login_id as phone_id
```

```
    INTO phone_id_tbl
```

```
    FROM source_table
```

```
    WHERE LENGTH(login_id) = 4;
```

```
    -- Create table with correctly formatted employee names
```

```
    SELECT DISTINCT (empno),
```

```
           SPLIT_PART(ename,',',2)|| ' ' || SPLIT_PART(ename,',',1) AS
```

```
ename
```

```
    INTO ename_split
```

```
    FROM source_table
```

```
    WHERE ename IS NOT NULL;
```

```
    -- Manager name table alteration from "LastName, FirstName" to
```

```
"FirstName LastName"
```

```
    SELECT DISTINCT (mgr),
```

```
           SPLIT_PART(mname,',',2)|| ' ' || SPLIT_PART(mname,',',1) AS
```

```
mname
```

```
    INTO mname_split
```

```
    FROM source_table
```

```
    WHERE mname IS NOT NULL;
```



```

-- Date table for hire date formatting and tenure calculation
SELECT DISTINCT (empno),
    AGE('2021-05-03', hiredate) as etenure,
    DATE_PART('day', hiredate) || '-' || date_part('month', hiredate) || '-'
|| date_part('year', hiredate) as hiredate
    INTO emp_tenure
FROM source_table;

```

-- Create new table for updated and cleaned data to be added

```

SELECT DISTINCT (e.empno),
    es.ename,
    e.job,
    e.mgr,
    ms.mname,
    et.hiredate,
    et.etenure,
    e.sal,
    e.commission,
    e.team_name,
    cht.chat_id,
    phn.phone_id

```

```

    INTO emp_clean
FROM emp as e
-- join chat_id_tbl to add chat_id column
LEFT JOIN chat_id_tbl as cht
ON e.empno = cht.empno
-- join phone_id_tbl to add phone_id column
LEFT JOIN phone_id_tbl as phn
ON e.empno = phn.empno
-- join ename_split to add fixed ename column
LEFT JOIN ename_split as es
ON e.empno = es.empno
-- join mname_split to add fixed mname column
LEFT JOIN mname_split as ms
ON e.mgr = ms.mgr
-- join emp_tenure
LEFT JOIN emp_tenure as et
ON e.empno = et.empno;

```

-- Delete temporary tables

```

DROP TABLE chat_id_tbl;
DROP TABLE phone_id_tbl;

```

```
DROP TABLE ename_split;
DROP TABLE mname_split;
DROP TABLE emp_tenure;
```

```
-- Fill in null values before syncing cleaned source_table with target_table
```

```
-- Populate null values with appropriate filler.
```

```
-- Numerical values are changed from "null" to "0"
```

```
UPDATE emp_clean
```

```
SET mgr = 0
```

```
WHERE
```

```
mgr IS NULL;
```

```
UPDATE emp_clean
```

```
SET commission = 0
```

```
WHERE
```

```
commission IS NULL;
```

```
-- string values are changed from "null" to "none."
```

```
UPDATE emp_clean
```

```
SET mname = 'none'
```

```
WHERE
```

```
mname IS NULL;
```

```
UPDATE emp_clean
```

```
SET chat_id = 'none'
```

```
WHERE
```

```
chat_id IS NULL;
```

```
-- Change the data types for clarity
```

```
ALTER TABLE emp_clean
```

```
ALTER COLUMN ename TYPE VARCHAR(100),
```

```
ALTER COLUMN mname TYPE VARCHAR(100);
```

```
-- Update target_table with cleaned source table data (emp_clean)
```

```
MERGE target_table AS tar
```

```
USING emp_clean AS src
```

```
ON src.empno = tar.empno
```

```
-- Identify rows to be updated when "empno" already exists in
```

```
target database and update if there are any changes
```

```
WHEN MATCHED
```

```
THEN UPDATE
```

```
SET ename = src.ename,
```

```
job = src.job,
```

```
mgr = src.mgr,
```

```

        mname = src.mname,
        hiredate = src.hiredate,
        etenure = src.etenure,
        sal = src.sal,
        commission = src.commission,
        team_name = src.team_name,
        chat_id = src.chat_id,
        phone_id = src.phone_id
-- Insert rows to be added that do not exist in target_table
WHEN NOT MATCHED
    THEN INSERT
        (empno,
         ename,
         job,
         mgr,
         mname,
         hiredate,
         etenure,
         sal,
         commission,
         team_name,
         chat_id,
         phone_id)
    VALUES
        (src.empno,
         src.ename,
         src.job,
         src.mgr,
         src.mname,
         src.hiredate,
         src.etenure,
         src.sal,
         src.commission,
         src.team_name,
         src.chat_id,
         src.phone_id);

-- Delete temporary table emp_clean
DROP TABLE emp_clean;

```

END

3. We want to rank employee salary values from our "EMP" table and then pivot the result set into three columns. Your task is to show the top 5, the next 5, then all the rest. You'll do this by ranking the employees by Salary and then pivot the results into three columns. Please be as detailed as possible for your answer.

-- There is no PIVOT function in PostgreSQL 11 and CROSSTAB function is not installed by default, therefore I created a new table to hold salary values to workaround the lack of functionality in my current platform.

-- Creates first "top_5" table by selecting the first 5 results from the SELECT query using row index

```
SELECT ROW_NUMBER() OVER (ORDER BY sal) + 0 AS row_num,  
       ename || ' (' || sal || ')' AS top_5  
INTO top_5  
FROM emp_clean_sorted  
ORDER BY sal DESC  
       OFFSET 0 ROWS  
       FETCH NEXT 5 ROWS ONLY;
```

	row_num bigint	top_5 text
1	14	Ted Mosbey (251643)
2	13	Hope Van Dyne (150986)
3	12	Chloe Sullivan (150986)
4	11	Alexander Luthor (149727)
5	10	Bruce Wayne (143436)

-- Creates the second "next_5" table by selecting the next 5 results using the OFFSET/FETCH parameters using row index plus the offset to align with top_5 table

```
SELECT ROW_NUMBER() OVER (ORDER BY sal) + 5 AS row_num,  
       ename || ' (' || sal || ')' AS next_5  
INTO next_5  
FROM emp_clean_sorted  
ORDER BY sal DESC  
       OFFSET 5 ROWS  
       FETCH NEXT 5 ROWS ONLY;
```

	row_num bigint		next_5 text	
1		14	Tony Stark (123305)	
2		13	Lana Lang (80526)	
3		12	Steve Rogers (75493)	
4		11	Bucky Barnes (65427)	
5		10	Pete Ross (62911)	





-- Creates the second "rest" table by selecting the rest of the results using the OFFSET/FETCH parameters using row index plus the offset to align with top_5 table

```
SELECT ROW_NUMBER() OVER (ORDER BY sal) + 10 AS row_num,
       ename || ' (' || sal || ')' AS rest
INTO rest
FROM emp_clean_sorted
ORDER BY sal DESC
      OFFSET 10 ROWS
;
```

	row_num bigint		rest text	
1		14	Clark Kent (62911)	
2		13	Ethan Hunt (55361)	
3		12	Jason Bourne (47812)	
4		11	Barney Stinson (40263)	

-- Uses JOIN to concatenate the top_5, next_5, and rest tables to mimic the pivot table shown as PIVOT is not available as a default function in PgAdmin 4 - Postgresql 11.

```
SELECT t5.top_5,
       n5.next_5,
       r.rest
INTO top_salaries
FROM top_5 AS t5
LEFT JOIN next_5 AS n5
ON t5.row_num = n5.row_num
LEFT JOIN rest AS r
ON t5.row_num = r.row_num
ORDER BY t5.row_num DESC;
```

	 top_5 text 	next_5 text 	rest text 
1	Ted Mosbey (251643)	Tony Stark (123305)	Clark Kent (62911)
2	Hope Van Dyne (150986)	Lana Lang (80526)	Ethan Hunt (55361)
3	Chloe Sullivan (150986)	Steve Rogers (75493)	Jason Bourne (47812)
4	Alexander Luthor (149727)	Bucky Barnes (65427)	Barney Stinson (40263)
5	Bruce Wayne (143436)	Pete Ross (62911)	[null]

4. Using a self-join write a query that counts how many employees report to each manager.

```
SELECT m.mname, COUNT(e.empno) as emp_Count
FROM emp_clean_sorted AS m, emp_clean_sorted AS e
WHERE m.mgr = e.empno
GROUP BY m.mname
ORDER BY emp_count DESC;
```

	mname character varying (100)	emp_count bigint
1	Ted Mosbey	5
2	Alexander Luthor	4
3	Bruce Wayne	3
4	Tony Stark	1

Typos in instructions:

- Header: misspells "Challenge" in SQL CODE Challenge
- Redundant "this" in the 1st bullet point in the instructions list in "Let's clean up..."
- Table used as example in question 3 in the "Top 5, Next 5, and rest" contains six entries in each column not 5 entries.