

# SQL Case study(SF Salaries)

29 February 2024 23:29 PM

Before uploading this data into SQL Server, basic analysis needs to be performed via Jupyter notebook

Below are some data cleaning steps that I will perform in SQL, although I have checked these mistakes using Python.

- Below are the table sample data
- The shape of the data is (148654, 13) (rows, columns).
- There is a column named 'Notes' with no values, so we need to delete this column. Some columns have missing values, and we will fill these values using the mean or median.
- There is a column where the **payable amount** mean is negative. We need to verify if this is possible because, according to my understanding, this should not happen.
- There is a columns **BasePay** where have some null and text values so need to clean this dataset
- Same problem we have in ['**OvertimePay**']
- In the '**Benefits**' column, we have multiple null values, so we need to drop this column.
- 74% values are missing in **status** column
- I will do feature's engineering, will make some columns for analysis
- Will make a **stored procedure** in SQL

1df.sample(10)

	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes	Agency	Status
7584	47585	Christopher Chambre	EMT/Paramedic/Firefighter	83216.52	13333.4	4312.9	28990.63	100862.82	129853.45	2012	NaN	San Francisco	NaN
4490	34491	JOHNNY ZHOU	TRANSIT OPERATOR	2529.28	0.0	0.0	NaN	2529.28	2529.28	2011	NaN	San Francisco	NaN
2301	122302	Steven A Norman	Deputy Probation Officer	95789.4	0.0	0.0	34800.84	95789.40	130590.24	2014	NaN	San Francisco	FT

1df.describe()

	Id	TotalPay	TotalPayBenefits	Year	Notes
count	148654.000000	148654.000000	148654.000000	148654.000000	0.0
mean	74327.500000	74768.321972	93692.554811	2012.522643	NaN
std	42912.857795	50517.005274	62793.533483	1.117538	NaN
min	1.000000	-618.130000	-618.130000	2011.000000	NaN
25%	37164.250000	36168.995000	44065.650000	2012.000000	NaN
50%	74327.500000	71426.610000	92404.090000	2013.000000	NaN
75%	111490.750000	105839.135000	132876.450000	2014.000000	NaN
max	148654.000000	567595.430000	567595.430000	2014.000000	NaN

1df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148654 entries, 0 to 148653
Data columns (total 13 columns):
#   Column              Non-Null Count  Dtype
---  -
0   Id                   148654 non-null  int64
1   EmployeeName         148654 non-null  object
2   JobTitle             148654 non-null  object
3   BasePay              148049 non-null  object
4   OvertimePay          148654 non-null  object
5   OtherPay             148654 non-null  object
6   Benefits             112495 non-null  object
7   TotalPay             148654 non-null  float64
8   TotalPayBenefits     148654 non-null  float64
9   Year                 148654 non-null  int64
10  Notes                0 non-null      float64
11  Agency               148654 non-null  object
12  Status               38119 non-null   object
dtypes: float64(3), int64(2), object(8)
memory usage: 14.7+ MB
```

- So I will upload this data on SQL and will perform some case studies

## Creating database

[create database sf;](#)

Uploading table manually in above created database

## Check some sample row and columns;

[SELECT TOP 10 \\* FROM Salaries ORDER BY NEWID\(\);](#)

Results		Messages												
	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Notes	Agency	Status	
1	49111	Danial Lemire	Deputy Court Clerk III	84512.31	0.0	1373.99	36720.51	85886.3	122606.81	2012	NULL	San Francisco	NULL	
2	45405	Sherman Hool	Registered Nurse	96669.62	3579.44	2430.16	39406.23	102679.22	142085.45	2012	NULL	San Francisco	NULL	
3	100674	Kenneth W Comier	Library Page	35554.47	0.0	378.77	17966.36	35933.24	53899.6	2013	NULL	San Francisco	NULL	
4	105009	Tonny Lukabya	StdntDsgn Train2/Arch/Eng/Plng	13853.94	0.0	0.0	90.2	13853.94	13944.14	2013	NULL	San Francisco	NULL	
5	13231	TOBY DYNER	PHYSICIAN SPECIALIST	82102.1	0.0	2025.0	NULL	84127.1	84127.1	2011	NULL	San Francisco	NULL	
6	28335	FRANCISCA AR...	HEALTH WORKER II	26312.32	0.0	910.0	NULL	27222.32	27222.32	2011	NULL	San Francisco	NULL	
7	136066	Albert F Sng	Library Page	48662.81	0.00	638.91	26332.76	49301.72	75634.48	2014	NULL	San Francisco	FT	
8	90534	Calvin F Watts	Transit Operator	67382.04	11358.03	5979.16	21335.61	84719.23	106054.84	2013	NULL	San Francisco	NULL	
9	74203	Sandy Feinland	Attorney (Civil/Criminal)	180051...	0.0	1250.0	49068.69	181301.01	230369.7	2013	NULL	San Francisco	NULL	
10	15564	PATRICK MARTI...	ESTATE INVESTIGATOR	75692.19	0.0	0.0	NULL	75692.19	75692.19	2011	NULL	San Francisco	NULL	

## Query to retrieve column information including descriptions

```
SELECT
COLUMN_NAME,
DATA_TYPE,
CHARACTER_MAXIMUM_LENGTH,
COLUMN_DEFAULT,
IS_NULLABLE,
COLUMNPROPERTY(object_id(TABLE_SCHEMA + '.' + TABLE_NAME), COLUMN_NAME, 'IsIdentity') AS IS_IDENTITY,
COLUMN_DESCRIPTION.value AS COLUMN_DESCRIPTION
FROM
INFORMATION_SCHEMA.COLUMNS
OUTER APPLY fn_listextendedproperty('MS_Description', 'SCHEMA', TABLE_SCHEMA, 'TABLE', TABLE_NAME, 'COLUMN', COLUMN_NAME) AS
COLUMN_DESCRIPTION
WHERE
TABLE_NAME = 'Salaries' -- Replace with your table name
ORDER BY
ORDINAL_POSITION;
```

## Output:

There are no default values and having null values in this dataset

Results		Messages					
	COLUMN_NAME	DATA_TYPE	CHARACTER_MAXIMUM_LENGTH	COLUMN_DEFAULT	IS_NULLABLE	IS_IDENTITY	COLUMN_DESCRIPTION
1	Id	nvarchar	50	NULL	YES	0	NULL
2	EmployeeName	nvarchar	50	NULL	YES	0	NULL
3	JobTitle	nvarchar	100	NULL	YES	0	NULL
4	BasePay	nvarchar	50	NULL	YES	0	NULL
5	OvertimePay	nvarchar	50	NULL	YES	0	NULL
6	OtherPay	nvarchar	50	NULL	YES	0	NULL
7	Benefits	nvarchar	50	NULL	YES	0	NULL
8	TotalPay	nvarchar	50	NULL	YES	0	NULL
9	TotalPayBenefits	nvarchar	50	NULL	YES	0	NULL
10	Year	nvarchar	50	NULL	YES	0	NULL
11	Notes	nvarchar	50	NULL	YES	0	NULL
12	Agency	nvarchar	50	NULL	YES	0	NULL
13	Status	nvarchar	50	NULL	YES	0	NULL

## Need to check total shape of the data

We have total number of rows(148654) and columns(13)

Rows

[select count\(\\*\) from Salaries](#)

## Columns

```
SELECT count(COLUMN_NAME)
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = 'Salaries';
```

### I will perform some data cleaning here:

- 1- There is a column named '**Notes**' with no values, so we need to delete this column. Some columns have missing values, and we will fill these values using the mean or median.

#### Recheck the col

```
select count(*) from Salaries
where Notes is null
```

#### -Drop a specific column from a table

```
ALTER TABLE Salaries
DROP COLUMN Notes;
```

- 2- There is a column where the **payable amount** mean is negative. We need to verify if this is possible because, according to my understanding, this should not happen.

I found that only one row has negative values, so I am deleting this particular row.  
After the row has been deleted, I intend to change the data type of this column to FLOAT

#### First check the numeric values

```
select * from(
    select round(cast(TotalPay as float),0)as total,
    case when round(cast(TotalPay as float),0) >0 then 1 else 0 end new_slab
    from Salaries)x
where x.new_slab=0
```

#### Delete this row (-618 this is the row)

```
delete from Salaries
where round(cast(TotalPay as float),0) =-618
```

#### now change the datatype of this col

```
ALTER TABLE Salaries
ALTER COLUMN TotalPay FLOAT;
```

#### Check the mean ,Min , Max ,Total for this col

```
SELECT
    AVG(TotalPay) AS Mean,
    min(TotalPay)as Min_values,
    max(TotalPay)as Max_values,
    sum(TotalPay)as Total_values,
    count(*)as total_count
FROM
    Salaries;
```

Mean	Min_values	Max_values	Total_values	total_count
74768.8291020801	0	567595.43	11114610752.5115	148653

- 3- There is a columns **BasePay** where have some null and text values so need to clean this dataset

There are 605 rows with null values. Additionally, some rows are filled with 'Not Provided,'

so I have decided to fill these rows with '0'."

```
select count(*) as total from Salaries
where BasePay is null
```

```
select BasePay from Salaries
order by BasePay desc
```

	BasePay
1	Not Provided
2	Not Provided
3	Not Provided
4	Not Provided
5	99998.02
6	99998.02
7	99998.01
8	99998.01
9	99990.65
10	99990.21

**Update query:**

```
UPDATE Salaries
SET BasePay=0
where BasePay ='Not Provided' or BasePay is null
```

**Change datatype**

```
ALTER TABLE Salaries
ALTER COLUMN BasePay FLOAT;
```

**check the mean ,Min ,Max ,Total for this col**

```
SELECT
    AVG(BasePay) AS Mean,
    min(BasePay) as Min_values,
    max(BasePay) as Max_values,
    sum(BasePay) as Total_values,
    count(*) as total_count
FROM
    Salaries;
```

	Mean	Min_values	Max_values	Total_values	total_count
1	66054.1801026068	0	319275.01	9819152034.7928	148653

4- Same problem we have in ['OvertimePay']

```
select OvertimePay from Salaries
order by OvertimePay asc
```

**Need to update some row where 'Not Provided' mentioned will replace by '0'**

```
UPDATE Salaries
SET OvertimePay=0
where OvertimePay ='Not Provided'
```

```
UPDATE Salaries
SET OvertimePay=0
where OvertimePay<0
```

**Change the datatype**

```
ALTER TABLE Salaries
ALTER COLUMN OvertimePay FLOAT;
```

**Check the mean ,Min ,Max , Total for this col**

```

SELECT
    AVG(OvertimePay) AS Mean,
    min(OvertimePay)as Min_values,
    max(OvertimePay)as Max_values,
    sum(OvertimePay)as Total_values,
    count(*)as total_count
FROM
    Salaries;

```

	Mean	Min_values	Max_values	Total_values	total_count
1	5065.95764720519	0	245131.88	753069802.129993	148653

5- In the 'Benefits' column, we have multiple null values, so we need to drop this column.

**Approximately 24% of the values in the Benefits column are null. I will fill these null values with '0'.**

```

select count(*) from Salaries
where Benefits is null

```

**Random Check**

```

select Benefits from Salaries
order by Benefits asc;

```

**Updating**

```

UPDATE Salaries
SET Benefits=0
where Benefits='Not Provided' or Benefits is null;

```

**Update <0 values into 0**

```

UPDATE Salaries
SET Benefits=0
where Benefits <0;

```

**Change the datatype**

```

ALTER TABLE Salaries
ALTER COLUMN Benefits FLOAT;

```

**Check the mean ,Min ,Max , Total for this col**

```

SELECT
    AVG(Benefits) AS Mean,
    min(Benefits)as Min_values,
    max(Benefits)as Max_values,
    sum(Benefits)as Total_values,
    count(*)as total_count
FROM
    Salaries;

```

	Mean	Min_values	Max_values	Total_values	total_count
1	18924.3601436233	-33.89	96570.66	2813162908.43003	148653

6- 74% values are missing in **status** column

```

select Status from Salaries
order by Status desc;

```

```

select Status,count(*)as total_cnt from Salaries
group by Status;

```

Update the all null values with 0

```
UPDATE Salaries
SET Status=0
where Status is null;
```

	Status	total_cnt
1	PT	15784
2	FT	22334
3	0	110535

7- I will do feature's engineering, will make some columns for analysis

## Features

1- TotalPay slab

2-TotalPayBenefits slab

3-TotalpayBenefits - Basepay>0 then 1 else 0 (How many emp getting overpay or benefits)

4-Designation columns (I am categorizing the 'Designation' column by counting the occurrences of each designation. If a designation appears more than 100 times, I will retain that designation; otherwise, I will label it as 'Other')

First Designation Columns:

```
ALTER TABLE Salaries
ADD Designation VARCHAR(255);
```

```
with main as (
  select lower(jobtitle) as jobtitle
  from salaries
  group by jobtitle
  having count(*) > 100
),
main2 as (
  select id,
  case when lower(jobtitle) in (select lower(jobtitle) from main) then lower(jobtitle) else 'others' end as designation_1
  from salaries
)
update salaries
set designation = main2.designation_1
from main2
where salaries.id = main2.id;
```

TotalPay slab Features/TotalPayBenefits

### Slab

```
0 to 10000 '0-10k'
10000 to 20000 '10k-20k'
20000 to 50000 '20-50k'
50000 to 100000 '50-1L'
>100000 '>=1L'
```

Create col

TotalPay

```
ALTER TABLE Salaries
ADD TotalPay_slab VARCHAR(255);
```

TotalPayBenefits

```
ALTER TABLE Salaries
ADD TotalPayBenefits_slab VARCHAR(255);
```

Updating columns:

```

With main as(
    select *,

        case when TotalPay between 0 and 10000 then '0-10k'
              when TotalPay between 10000 and 20000 then '10k-20K'
              when TotalPay between 20001 and 50000 then '20K-50k'
              when TotalPay between 50001 and 100000 then '50K-1L'
              else 'Above 1L'
        end TotalPay_slab_temp,

        case when TotalPayBenefits between 0 and 10000 then '0-10k'
              when TotalPayBenefits between 10000 and 20000 then '10k-20K'
              when TotalPayBenefits between 20001 and 50000 then '20K-50k'
              when TotalPayBenefits between 50001 and 100000 then '50K-1L'
              else 'Above 1L'
        end TotalPayBenefits_temp

    from Salaries)
UPDATE Salaries
SET TotalPay_slab=main.TotalPay_slab_temp,
    TotalPayBenefits_slab=main.TotalPayBenefits_temp
from main
where main.id=Salaries.id

```

### Output

		BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year	Agency	Status	Designation	TotalPay_slab	TotalPayBenefits_slab
1	IN TRANSIT AUTHORITY	167411.18	0	400184.25	0	567595.43	567595.43	2011	San Francisco	0	others	Above 1L	Above 1L
2		155966.02	245131.88	137811.38	0	538909.28	538909.28	2011	San Francisco	0	others	Above 1L	Above 1L
3		212739.13	106088.18	16452.6	0	335279.91	335279.91	2011	San Francisco	0	others	Above 1L	Above 1L
4	MECHANIC	77916	56120.71	198306.9	0	332343.61	332343.61	2011	San Francisco	0	others	Above 1L	Above 1L
5	IRE DEPARTMENT)	134401.6	9737	182234.59	0	326373.19	326373.19	2011	San Francisco	0	others	Above 1L	Above 1L
6		118602	8601	189082.74	0	316285.74	316285.74	2011	San Francisco	0	others	Above 1L	Above 1L
7	ENT)	92492.01	89062.9	134426.14	0	315981.05	315981.05	2011	San Francisco	0	others	Above 1L	Above 1L
8	TS	256576.96	0	51322.5	0	307899.46	307899.46	2011	San Francisco	0	others	Above 1L	Above 1L

### TotalpayBenefits - Basepay>0 then 1 else 0 (How many emp getting overpay or benefits)

#### Columns adding

```

ALTER TABLE Salaries
ADD BasePay_TotalpayBenefits int

```

#### Updating col

```

with main as(
    select *,
        case when (TotalPayBenefits-BasePay)>=1 then 1 else 0 end as TotalpayBenefits_temp
    from Salaries)
UPDATE Salaries
SET BasePay_TotalpayBenefits=main.TotalpayBenefits_temp
from main
where Salaries.id=main.id;

```

```

select * from Salaries;

```



### I Have clean this data and the final output below:



	Id	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay	TotalPayBenefits	Year
1	1	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411.18	0	400184.25	0	567595.43	567595.4375	2011
2	2	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966.02	245131.88	137811.38	0	538909.28	538909.25	2011
3	3	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739.13	106088.18	16452.6	0	335279.91	335279.90625	2011
4	4	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE MECHANIC	77916	56120.71	198306.9	0	332343.61	332343.625	2011
5	5	PATRICK GARDNER	DEPUTY CHIEF OF DEPARTMENT,(FIRE DEPARTMENT)	134401.6	9737	182234.59	0	326373.19	326373.1875	2011
6	6	DAVID SULLIVAN	ASSISTANT DEPUTY CHIEF II	118602	8601	189082.74	0	316285.74	316285.75	2011
7	7	ALISON IFF	BATTALION CHIEF (FIRE DEPARTMENT)	92492.01	89062.9	134426.14	0	315981.05	315981.0625	2011

Agency	Status	Designation	TotalPay_slab	TotalPayBenefits_slab	BasePay_TotalpayBenefits
San Francisco	0	others	Above 1L	Above 1L	1
San Francisco	0	others	Above 1L	Above 1L	1
San Francisco	0	others	Above 1L	Above 1L	1
San Francisco	0	others	Above 1L	Above 1L	1
San Francisco	0	others	Above 1L	Above 1L	1
San Francisco	0	others	Above 1L	Above 1L	1
San Francisco	0	others	Above 1L	Above 1L	1

## Stored procedure

- I will create some stored procedure:

I will create a query where I will pass a variable for 'top 10' or 'top 15,' and the query will return results accordingly

**highest-paying jobs for each year (Job title, designation, total pay slab, total pay amount)**

```
CREATE PROCEDURE top_rows (@top int)
AS
BEGIN
with main as(
    select EmployeeName,
           Designation,
           TotalPay,
           Year,
           TotalPay_slab,
           ROW_NUMBER()over(partition by Year order by TotalPay)as rn
    from Salaries)
select * from main
where main.rn<= @top
END;
```

**EXEC top\_rows 1** --- The procedure will return values based on the provided argument, such as top 10, 1, or 2.

When I give the 1 in argument the is below:

	EmployeeName	Designation	TotalPay	Year	TotalPay_slab	m
1	NATHANIEL FORD	others	567595.43	2011	Above 1L	1
2	Gary Altenberg	lieutenant, fire suppression	362844.66	2012	Above 1L	1
3	Samson Lai	others	347102.32	2013	Above 1L	1
4	David Shinn	others	471952.64	2014	Above 1L	1

When I give the 5 in argument the is below:

It will give me top 5 rows for every year with the high paying job with other fields



	EmployeeName	Designation	TotalPay	Year	TotalPay_slab	m
1	NATHANIEL FORD	others	567595.43	2011	Above 1L	1
2	GARY JIMENEZ	others	538909.28	2011	Above 1L	2
3	ALBERT PARDINI	others	335279.91	2011	Above 1L	3
4	CHRISTOPHER CHONG	others	332343.61	2011	Above 1L	4
5	PATRICK GARDNER	others	326373.19	2011	Above 1L	5
6	Gary Altenberg	lieutenant, fire suppression	362844.66	2012	Above 1L	1
7	John Goldberg	captain 3	350403.41	2012	Above 1L	2
8	Frederick Binkley	emt/paramedic/firefighter	337204.86	2012	Above 1L	3
9	Khoa Trinh	electronic maintenance tech	336393.73	2012	Above 1L	4
10	Mark Kearney	others	327689.78	2012	Above 1L	5
11	Samson Lai	others	347102.32	2013	Above 1L	1

Create a stored procedure that retrieves the top 5 highest-paying designation roles for a specified year (use a variable for the year).

```
CREATE PROCEDURE role_high (@year varchar(255))
AS
BEGIN
with main as(
    select Designation,TotalPay,
        ROW_NUMBER()over(partition by Designation order by TotalPay desc)as rn
    from Salaries
    where Year=@year)
select * from main
where main.rn<=5

END;

EXEC role_high 2012
```

	Designation	TotalPay	m
1	account clerk	63194.81	1
2	account clerk	58399.63	2
3	account clerk	57684.62	3
4	account clerk	57133.52	4
5	account clerk	55504.94	5
6	accountant ii	76715.07	1
7	accountant ii	75179.98	2
8	accountant ii	74182.9	3
9	accountant ii	74165.5	4
10	accountant ii	74165.47	5
11	accountant iii	103766.03	1

I've completed data cleaning, performed feature engineering, and applied stored procedures to enhance a dataset, optimizing it for analysis and modeling