

## df\_dept



-----  
df\_dept  
-----

department_id	department_name	location_id
20	Marketing	180
30	Purchasing	1700
40	Human Resources	2400
50	Shipping	1500
60	IT	1400
70	Public Relations	2700
80	Sales	2500
90	Executive	1700
100	Finance	1700
110	Accounting	1700
120	Treasury	1700
130	Corporate Tax	1700
140	Control And Credit	1700
150	Shareholder Services	1700
160	Benefits	1700
170	Payroll	1700

# df\_employees



```
-- df_employees --
```

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	commission_pct	manager_id	department_id
100	Steven	King	SKING	515.123.4567	1987-06-17	AD_PRES	24000	null	null	20
101	Neena	Kochhar	NKOCHHAR	515.123.4568	1989-11-21	AD_VP	17000	null	100	20
102	Lex	De Haan	LDEHAAN	515.123.4569	1993-09-12	AD_VP	17000	null	100	30
103	Alexander	Hunold	AHUNOLD	590.423.4567	1990-09-30	IT_PROG	9000	null	102	60
104	Bruce	Ernst	BERNST	590.423.4568	1991-05-21	IT_PROG	6000	null	103	60
105	David	Austin	DAUSTIN	590.423.4569	1997-06-25	IT_PROG	4800	null	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	1998-02-05	IT_PROG	4800	null	103	40
107	Diana	Lorentz	DLORENTZ	590.423.5567	1999-02-09	IT_PROG	4200	null	103	40
108	Nancy	Greenberg	NGREENBE	515.124.4569	1994-08-17	FI_MGR	12000	null	101	100
109	Daniel	Faviet	DFAVIET	515.124.4169	1994-08-12	FI_ACCOUNT	9000	null	108	170
110	John	Chen	JCHEN	515.124.4269	1997-04-09	FI_ACCOUNT	8200	null	108	170
111	Ismael	Sciarra	ISCIARRA	515.124.4369	1997-02-01	FI_ACCOUNT	7700	null	108	160
112	Jose Manuel	Urman	JMURMAN	515.124.4469	1998-06-03	FI_ACCOUNT	7800	null	108	150
113	Luis	Popp	LPOPP	515.124.4567	1999-12-07	FI_ACCOUNT	6900	null	108	140
114	Den	Raphaely	DRAPHEAL	515.127.4561	1994-11-08	PU_MAN	11000	null	100	30
115	Alexander	Khoo	AKHOO	515.127.4562	1995-05-12	PU_CLERK	3100	null	114	80
116	Shelli	Baida	SBAIDA	515.127.4563	1997-12-13	PU_CLERK	2900	null	114	70
117	Sigal	Tobias	STOBIAZ	515.127.4564	1997-09-10	PU_CLERK	2800	null	114	30
118	Guy	Himuro	GHIMURO	515.127.4565	1998-01-02	PU_CLERK	2600	null	114	60
119	Karen	Colmenares	KCOLMENA	515.127.4566	1999-04-08	PU_CLERK	2500	null	114	130
120	Matthew	Weiss	MWEISS	650.123.1234	1996-07-18	ST_MAN	8000	null	100	50
121	Adam	Fripp	AFRIPP	650.123.2234	1997-08-09	ST_MAN	8200	null	100	50
122	Payam	Kaufling	PKAUFLIN	650.123.3234	1995-05-01	ST_MAN	7900	null	100	40
123	Shanta	Vollman	SVOLLMAN	650.123.4234	1997-10-12	ST_MAN	6500	null	100	50
124	Kevin	Mourgos	KMOURGOS	650.123.5234	1999-11-12	ST_MAN	5800	null	100	80
125	Julia	Nayer	JNAYER	650.124.1214	1997-07-02	ST_CLERK	3200	null	120	50
126	Irene	Mikkilineni	IMIKKILI	650.124.1224	1998-11-12	ST_CLERK	2700	null	120	50
127	James	Landry	JLANDRY	650.124.1334	1999-01-02	ST_CLERK	2400	null	120	90
128	Steven	Markle	SMARKLE	650.124.1434	2000-03-04	ST_CLERK	2200	null	120	50
129	Laura	Bissot	LBISSOT	650.124.5234	1997-09-10	ST_CLERK	3300	null	121	50
130	Mozhe	Atkinson	MATKINS0	650.124.6234	1997-10-12	ST_CLERK	2800	null	121	110

# 1 - Select employee's first name, last name, job\_id, and salary whose first name starts with alphabet 'S'

```
result = (  
    df_employees.filter(F.col('first_name')  
        .like('S%'))  
    .select('first_name', 'last_name', 'job_id', 'salary')  
).show()
```



first_name	last_name	job_id	salary
Steven	King	AD_PRES	24000
Shelli	Baida	PU_CLERK	2900
Sigal	Tobias	PU_CLERK	2800
Shanta	Vollman	ST_MAN	6500
Steven	Markle	ST_CLERK	2200

## 2 - Find the employees who joined the company after 15th of the month

```
result = (
    df_employees.filter(F.dayofmonth('hire_date') >= 15)
    .select('first_name', 'last_name', 'hire_date')
).show()
```

```
+-----+-----+-----+
|first_name|last_name| hire_date|
+-----+-----+-----+
|      Steven|       King|1987-06-17|
|      Neena|    Kochhar|1989-11-21|
| Alexander|     Hunold|1990-09-30|
|      Bruce|      Ernst|1991-05-21|
|      David|      Austin|1997-06-25|
|     Nancy|Greenberg|1994-08-17|
|    Matthew|      Weiss|1996-07-18|
+-----+-----+-----+
```

### 3 - Display the employee's first name and first name in reverse order

```
result = (
    df_employees.select('first_name', F.reverse('first_name')
        .alias('name reversed'))
).show(10)
```

```
+-----+-----+
|first_name|name reversed|
+-----+-----+
|      Steven|      nevetS|
|      Neena|      aneeN|
|       Lex|      xeL|
| Alexander|      rednaxelA|
|      Bruce|      ecurB|
|      David|      divaD|
|      Valli|      illaV|
|      Diana|      anaiD|
|      Nancy|      ycnaN|
|     Daniel|      leinaD|
+-----+-----+
only showing top 10 rows
```

## 4 - Display the 5 least earning employees

```
result = (
    df_employees.select('first_name', 'last_name', 'salary')
        .orderBy(F.asc('salary'))
        .limit(5)
).show()
```



first_name	last_name	salary
Steven	Markle	2200
James	Landry	2400
Karen	Colmenares	2500
Guy	Himuro	2600
Irene	Mikkilineni	2700

## 5 - Find the employees hired in the 80s

```
result = (  
    df_employees.select('first_name', 'last_name', 'hire_date')  
        .filter(F.year('hire_date')  
            .between(1980, 1989))  
).show()
```



first_name	last_name	hire_date
Steven	King	1987-06-17
Neena	Kochhar	1989-11-21

## 6 - Find the maximum salary from each department.

```
joined_df = df_employees.join(df_dept,df_employees.department_id == df_dept.department_id, 'left')

result = (
    joined_df.groupBy('department_name')
    .agg(F.max('salary').alias('dept max salary'))
).show()
```

department_name	dept max salary
Purchasing	17000
Marketing	24000
IT	9000
Payroll	9000
Finance	12000
Benefits	7700
Human Resources	7900
Shareholder Services	7800
Control And Credit	6900
Sales	5800
Public Relations	2900
Corporate Tax	2500
Shipping	8200
Accounting	2800
Executive	2400

## 7 - Find employees who earn more than the average salary in that company

```
average_salary = df_employees.select(F.avg('salary').alias("avg_salary")).first()["avg_salary"]

result = (
    df_employees.filter(F.col('salary') > average_salary)
        .select('first_name', 'last_name', 'salary')
).show()
```

first_name	last_name	salary
Steven	King	24000
Neena	Kochhar	17000
Lex	De Haan	17000
Alexander	Hunold	9000
Nancy	Greenberg	12000
Daniel	Faviet	9000
John	Chen	8200
Ismael	Sciarra	7700
Jose Manuel	Urman	7800
Den	Raphaely	11000
Matthew	Weiss	8000
Adam	Fripp	8200
Payam	Kaufling	7900

## 8 - Find the employees who joined in August, 1994.

```
result = (  
    df_employees.filter((F.year('hire_date') == 1994) & (F.month('hire_date') == 8))  
        .select(F.concat('first_name', F.lit(' '), 'last_name').alias('emp_name'),  
'hire_date')  
).show( )
```

emp_name	hire_date
Nancy Greenberg	1994-08-17
Daniel Faviet	1994-08-12

## 9 - Select employee first name and the corresponding phone number in the format (\_ \_ )-( \_ )-( \_ \_ )



```
result = (
    df_employees.select(F.concat('first_name'),
F.lit(' '), 'last_name').alias('emp name'),
F regexp_replace('phone_number', '\.', '-').
alias('phn number'))
).show(10)
```



emp name	phn number
Steven King	515-123-4567
Neena Kochhar	515-123-4568
Lex De Haan	515-123-4569
Alexander Hunold	590-423-4567
Bruce Ernst	590-423-4568
David Austin	590-423-4569
Valli Pataballa	590-423-4560
Diana Lorentz	590-423-5567
Nancy Greenberg	515-124-4569
Daniel Faviet	515-124-4169

only showing top 10 rows

## 10- Select the employees whose first\_name contains “an”

```
result = (  
    df_employees.filter(F.col('first_name').like('%an%'))  
    .select('first_name', 'last_name')  
).show()
```

first_name	last_name
Alexander	Hunold
Diana	Lorentz
Nancy	Greenberg
Daniel	Faviet
Jose Manuel	Urman
Alexander	Khoo
Shanta	Vollman

# 11- Write a query to divide people into three groups based on their salaries



```
# Define the salary thresholds for the three groups
thresholds = [2000, 5000, 10000]

# Create a User-Defined Function (UDF) to categorize salaries into
groups
def categorize_salary(salary):
    if salary >= thresholds[0] and salary < thresholds[1]:
        return 'low'
    elif salary >= thresholds[1] and salary < thresholds[2]:
        return 'mid'
    else:
        return 'high'

# Register the UDF
categorize_salary_udf = F.udf(categorize_salary, StringType())

# Add a new column to the DataFrame indicating the salary group
df_employees = df_employees.withColumn("salary_group",
categorize_salary_udf(F.col('salary')))

# Show the DataFrame with the new column
df_employees.select('first_name', 'last_name', 'salary',
'salary_group', 'salary_group').orderBy('salary').show( )
```



first_name	last_name	salary	salary_group	salary_group
Steven	Markle	2200	low	low
James	Landry	2400	low	low
Karen	Colmenares	2500	low	low
Guy	Himuro	2600	low	low
Irene	Mikkilineni	2700	low	low
Sigal	Tobias	2800	low	low
Mozhe	Atkinson	2800	low	low
Shelli	Baida	2900	low	low
Alexander	Khoo	3100	low	low
Julia	Nayer	3200	low	low
Laura	Bissot	3300	low	low
Diana	Lorentz	4200	low	low
David	Austin	4800	low	low
Valli	Pataballa	4800	low	low
Kevin	Mourgos	5800	mid	mid
Bruce	Ernst	6000	mid	mid
Shanta	Vollman	6500	mid	mid
Luis	Popp	6900	mid	mid
Ismael	Sciarra	7700	mid	mid
Jose Manuel	Urman	7800	mid	mid

only showing top 20 rows

## 12 - Find the salary range of employees



```
result = (
    df_employees.select(F.max('salary').alias('max salary'),
                        F.min('salary').alias('min salary'),
                        F.round(F.avg('salary')).alias('avg salary'))
).show()
```



max salary	min salary	avg salary
24000	2200	6977.0

## 13 - Get the count of employees hired year wise



```
result = (
    df_employees.groupBy(F.year('hire_date').alias('hire year'))
        .agg(F.count('employee_id').alias('employee count'))
        .orderBy('hire year')
).show()
```



hire year	employee count
1987	1
1989	1
1990	1
1991	1
1993	1
1994	3
1995	2
1996	1
1997	10
1998	4
1999	5
2000	1

## 14 - Find the count of employees in each department



```
joined_df = df_dept.join(df_employees, df_employees['department_id'] == df_dept['department_id'], 'left')

result = (
    joined_df.groupBy(df_dept['department_name'])
        .agg(F.count(df_employees['employee_id']).alias('emp_count'))
        .orderBy(F.desc('emp_count'))
).show()
```



department_name	emp_count
Shipping	7
IT	4
Purchasing	3
Human Resources	3
Marketing	2
Sales	2
Payroll	2
Public Relations	1
Executive	1
Accounting	1
Finance	1
Corporate Tax	1
Shareholder Services	1
Control And Credit	1
Benefits	1
Treasury	0

# 15- Find count of employees under each manager in descending order



```
# Alias for the first instance of the DataFrame
emp_df = df_employees.alias("emp_df")

# Alias for the second instance of the DataFrame
mgr_df = df_employees.alias("mgr_df")

joined_df = mgr_df.join(emp_df, F.col('emp_df.manager_id') == F.col('mgr_df.employee_id'), 'inner')

result = (
    joined_df.groupBy(F.concat('mgr_df.first_name', F.lit(' '), 'mgr_df.last_name').alias('manager'))
        .agg(F.count(F.col('emp_df.employee_id')).alias('reportee_count'))
        .orderBy(F.desc('reportee_count'))
).show()
```



manager	reportee_count
Steven King	8
Nancy Greenberg	5
Den Raphaely	5
Matthew Weiss	4
Alexander Hunold	4
Adam Fripp	2
Lex De Haan	1
Neena Kochhar	1

# 16 - Display employees and their corresponding managers and with their salaries

```
# Alias for the first instance of the DataFrame  
emp_df = df_employees.alias("emp_df")  
  
# Alias for the second instance of the DataFrame  
mgr_df = df_employees.alias("mgr_df")  
  
joined_df = emp_df.join(mgr_df, F.col('emp_df.manager_id') == F.col('mgr_df.employee_id'), 'inner')  
  
result = (  
    joined_df.select(F.concat('emp_df.first_name', F.lit(' '), 'emp_df.last_name').alias('emp_name'),  
    F.col('emp_df.salary').alias('emp_salary'), F.concat('mgr_df.first_name', F.lit(' '),  
    'mgr_df.last_name').alias('manager_name'), F.col('mgr_df.salary').alias('manager_salary'))  
).show(10)
```



emp_name	emp_salary	manager_name	manager_salary
Neena Kochhar	17000	Steven King	24000
Lex De Haan	17000	Steven King	24000
Den Raphaely	11000	Steven King	24000
Matthew Weiss	8000	Steven King	24000
Adam Fripp	8200	Steven King	24000
Payam Kaufling	7900	Steven King	24000
Shanta Vollman	6500	Steven King	24000
Kevin Mourgos	5800	Steven King	24000
Nancy Greenberg	12000	Neena Kochhar	17000
Alexander Hunold	9000	Lex De Haan	17000

only showing top 10 rows

## 17 - Display employees first name, last name, job\_id and salary whose first name starts with alphabet S



```
result = (
    df_employees.filter(F.col('first_name')
        .like('S%'))
    .select('first_name', 'last_name', 'job_id', 'salary')
).show( )
```



first_name	last_name	job_id	salary
Steven	King	AD_PRES	24000
Shelli	Baida	PU_CLERK	2900
Sigal	Tobias	PU_CLERK	2800
Shanta	Vollman	ST_MAN	6500
Steven	Markle	ST_CLERK	2200

## 18 - Find employee with the highest salary

```
windowSpec = Window.orderBy(F.desc('salary'))  
  
result = (  
    df_employees.withColumn('rnk', F.dense_rank().over(windowSpec))  
        .filter(F.col('rnk') == 1)  
        .select('employee_id', 'first_name', 'last_name', 'salary')  
).show()
```



employee_id	first_name	last_name	salary
100	Steven	King	24000

- 19 - Fetch employees with the nth highest salary



```
def get_top_ranked_employees(rank):
    windowSpec = Window.orderBy(F.desc('salary'))
    result = (
        df_employees.withColumn('rnk', F.dense_rank().over(windowSpec))
        .filter(F.col('rnk') == rank)
        .select('employee_id', 'first_name', 'last_name', 'salary')
    )
    return result

# Example: Get the second-highest salary employee
rank_2_result = get_top_ranked_employees(2)
rank_2_result.show()

# Example: Get the third-highest salary employee
rank_3_result = get_top_ranked_employees(3)
rank_3_result.show()
```



2nd highest salary:

```
+-----+-----+-----+-----+
|employee_id|first_name|last_name|salary|
+-----+-----+-----+-----+
|          101|      Neena|  Kochhar| 17000|
|          102|        Lex| De Haan| 17000|
+-----+-----+-----+-----+
```

## • 20 - Find the managers and the reporting employees who work in different departments

```
# Rename columns in the first DataFrame
emp_df = (
    df_employees.withColumnRenamed("department_id", "emp_department_id")
        .withColumnRenamed('first_name', 'emp_first_name')
        .withColumnRenamed('last_name', 'emp_last_name')
        .withColumnRenamed('department_id', 'emp_department_id')
).alias('emp_df')

# Rename columns in the second DataFrame
mgr_df = (
    df_employees.withColumnRenamed("department_id", "mgr_department_id")
        .withColumnRenamed('first_name', 'mgr_first_name')
        .withColumnRenamed('last_name', 'mgr_last_name')
        .withColumnRenamed('department_id', 'mgr_department_id')
).alias('mgr_df')

# Join the DataFrames and filter using qualified column names
joined_df = emp_df.join(mgr_df, F.col("emp_df.manager_id") == F.col("mgr_df.employee_id"), "inner")

# Specify the DataFrame alias when referencing the columns in the filter condition
filtered_df = (
    joined_df.filter(emp_df["emp_department_id"] != mgr_df["mgr_department_id"])
    .select(F.concat(mgr_df['mgr_first_name'], F.lit(' '), mgr_df['mgr_last_name']).alias('manager_name'),
    mgr_df['mgr_department_id'], F.concat(emp_df['emp_first_name'], F.lit(' '), emp_df['emp_last_name']).alias('emp_name'),
    emp_df['emp_department_id'])
)
display(filtered_df)
```



```
+-----+-----+-----+-----+
|manager_name|mgr_department_id|      emp name|emp_department_id|
+-----+-----+-----+-----+
| Steven King|          20|    Lex De Haan|          30|
| Steven King|          20| Den Raphaely|          30|
| Steven King|          20| Matthew Weiss|          50|
| Steven King|          20|     Adam Fripp|          50|
| Steven King|          20|Payam Kaufling|          40|
+-----+-----+-----+-----+
only showing top 5 rows
```