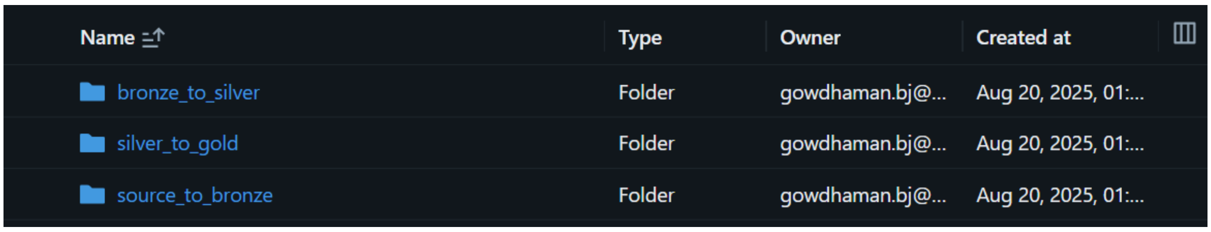
Databricks Assignment

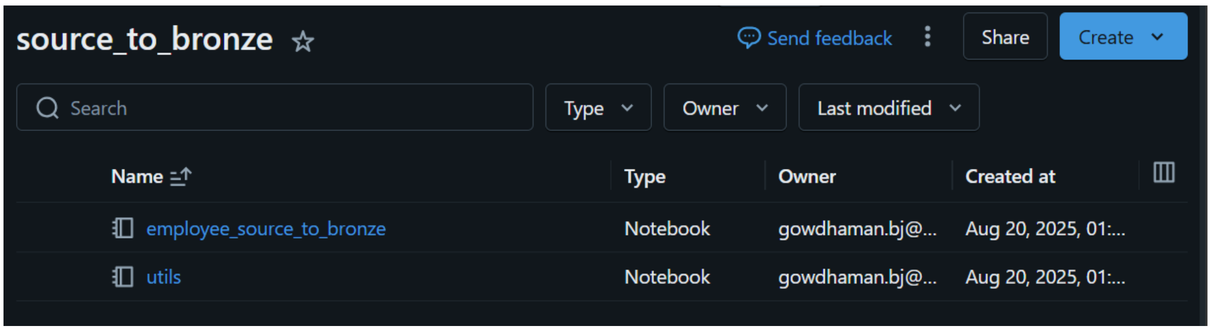
Question 1:

1. Create 3 folders as source_to_bronze, bronze_to_silver, silver_to_gold.



Name	Type	Owner	Created at
bronze_to_silver	Folder	gowdhaman.bj@...	Aug 20, 2025, 01:...
silver_to_gold	Folder	gowdhaman.bj@...	Aug 20, 2025, 01:...
source_to_bronze	Folder	gowdhaman.bj@...	Aug 20, 2025, 01:...

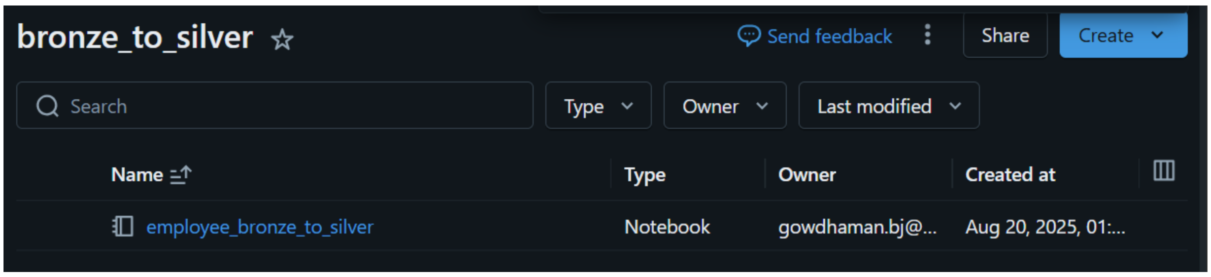
2. Create 4 notebooks in this respective order.
   2 Notebooks named in source_to_bronze as utils (add all common functions in this notebook) and employee_source_to_bronze (driver notebook)



source_to_bronze

Name	Type	Owner	Created at
employee_source_to_bronze	Notebook	gowdhaman.bj@...	Aug 20, 2025, 01:...
utils	Notebook	gowdhaman.bj@...	Aug 20, 2025, 01:...

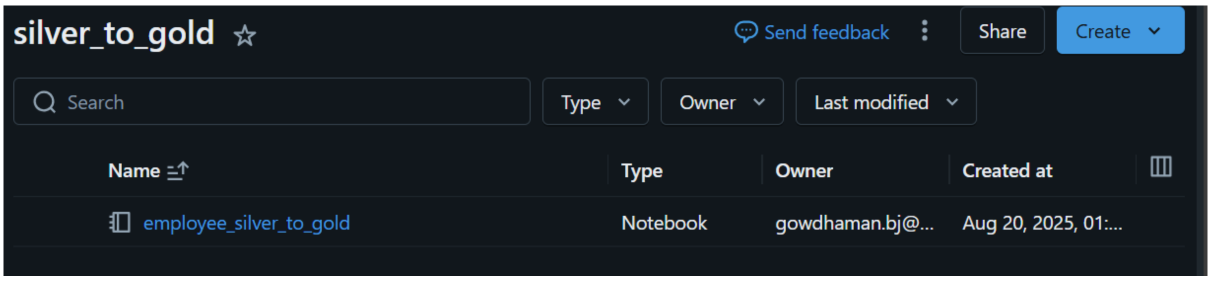
   1 Notebook in bronze to silver as employee_bronze_to_silver



bronze_to_silver

Name	Type	Owner	Created at
employee_bronze_to_silver	Notebook	gowdhaman.bj@...	Aug 20, 2025, 01:...

   1 Notebook in silver to gold as employee_silver_to_gold

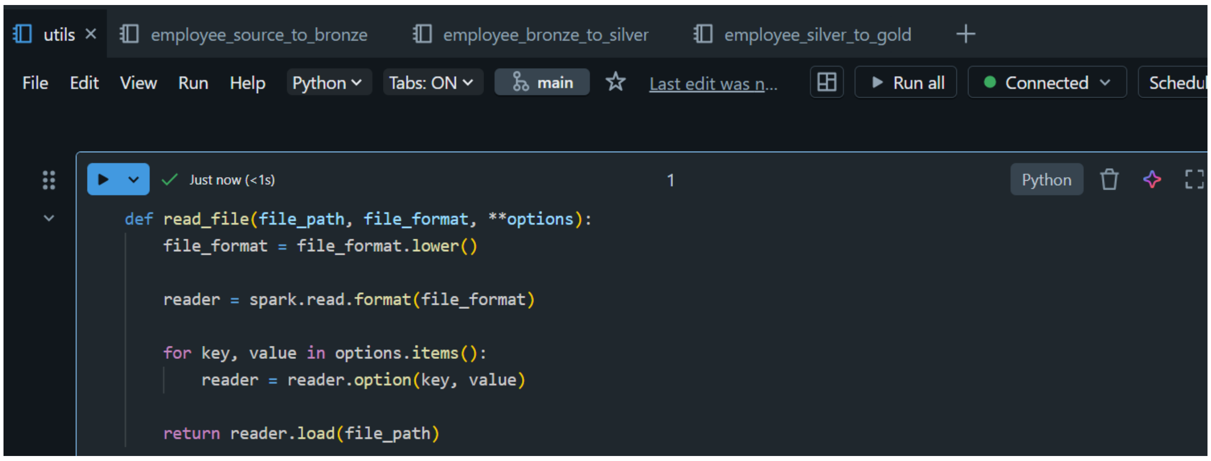


silver_to_gold

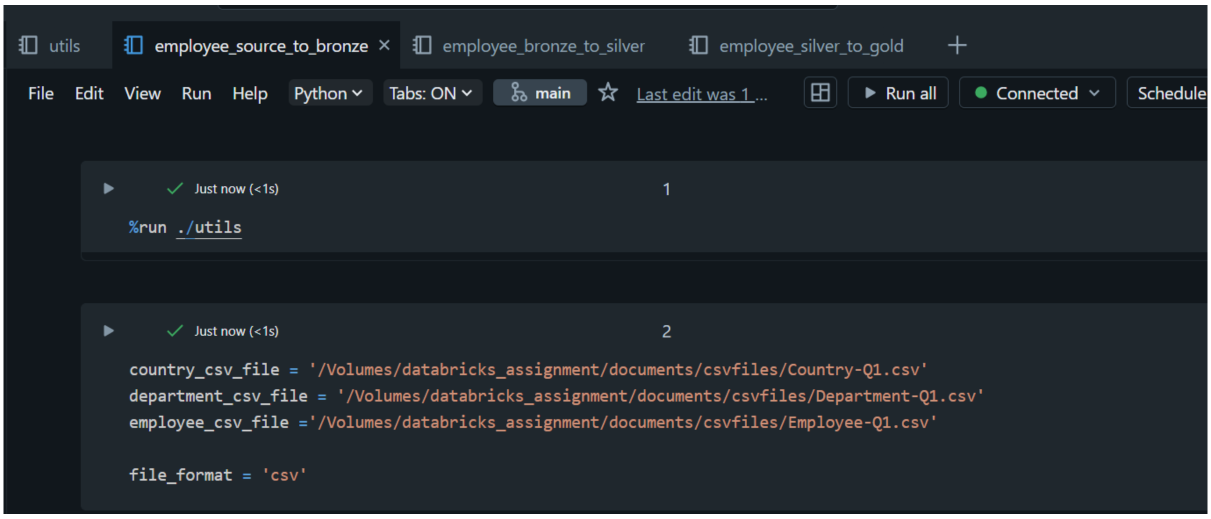
Name	Type	Owner	Created at
employee_silver_to_gold	Notebook	gowdhaman.bj@...	Aug 20, 2025, 01:...

3. Read the 3 datasets as Dataframe in **employee_source_to_bronze**, call utils notebook in this notebook, and write to a location in DBFS, as

/source_to_bronze/file_name.csv (employee, department_df, country_df) as CSV format.

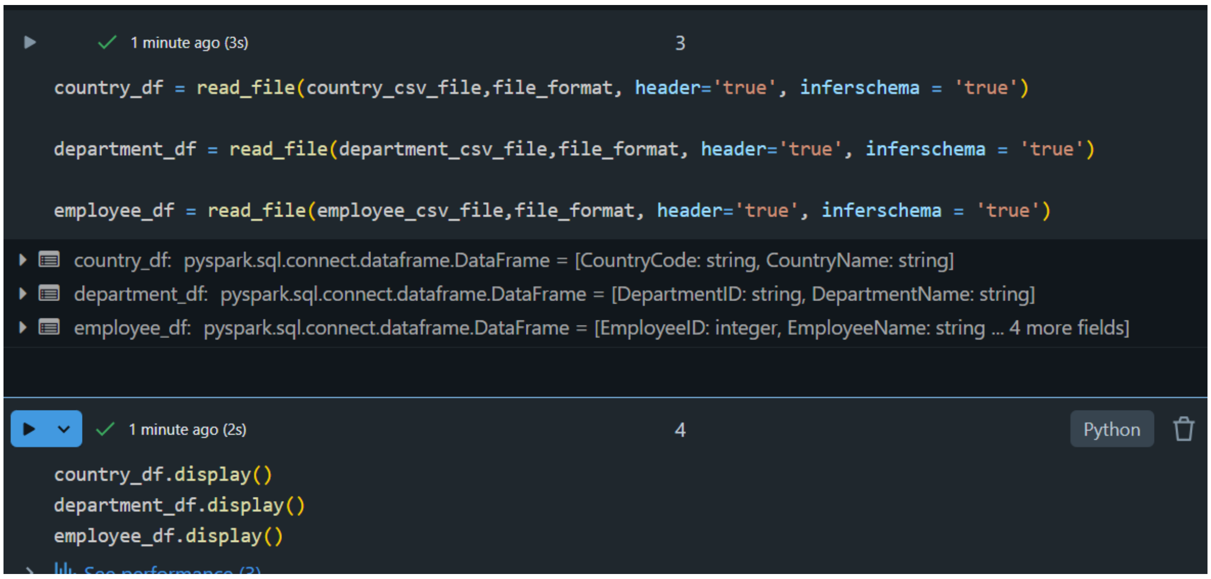

utils
employee_source_to_bronze
employee_bronze_to_silver
employee_silver_to_gold
File Edit View Run Help Python Tabs: ON main Last edit was n... Run all Connected Schedul
Just now (<1s)
1
Python

```python
def read_file(file_path, file_format, **options):
    file_format = file_format.lower()

    reader = spark.read.format(file_format)

    for key, value in options.items():
        reader = reader.option(key, value)

    return reader.load(file_path)
```


utils
employee_source_to_bronze
employee_bronze_to_silver
employee_silver_to_gold
File Edit View Run Help Python Tabs: ON main Last edit was 1 ... Run all Connected Schedule
Just now (<1s)
1

```
%run ./utils
```

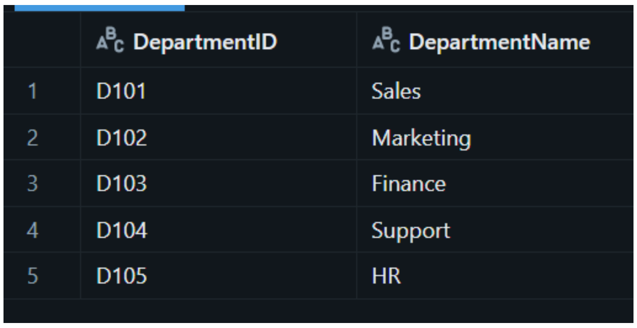
```python
country_csv_file = '/Volumes/databricks_assignment/documents/csvfiles/Country-Q1.csv'
department_csv_file = '/Volumes/databricks_assignment/documents/csvfiles/Department-Q1.csv'
employee_csv_file ='/Volumes/databricks_assignment/documents/csvfiles/Employee-Q1.csv'

file_format = 'csv'
```



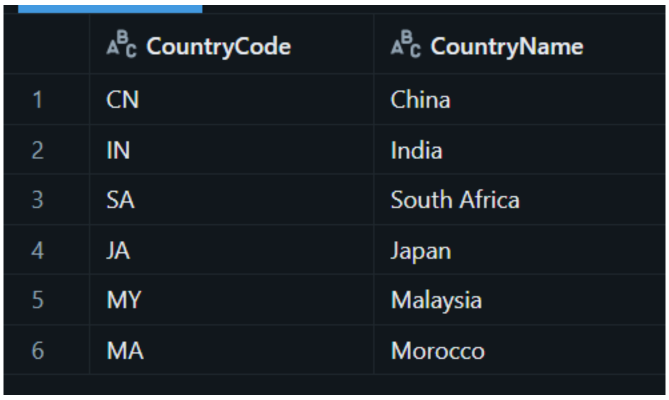
```python
country_df = read_file(country_csv_file,file_format, header='true', inferschema = 'true')

department_df = read_file(department_csv_file,file_format, header='true', inferschema = 'true')

employee_df = read_file(employee_csv_file,file_format, header='true', inferschema = 'true')
```

```
country_df: pyspark.sql.connect.dataframe.DataFrame = [CountryCode: string, CountryName: string]
department_df: pyspark.sql.connect.dataframe.DataFrame = [DepartmentID: string, DepartmentName: string]
employee_df: pyspark.sql.connect.dataframe.DataFrame = [EmployeeID: integer, EmployeeName: string ... 4 more fields]
```

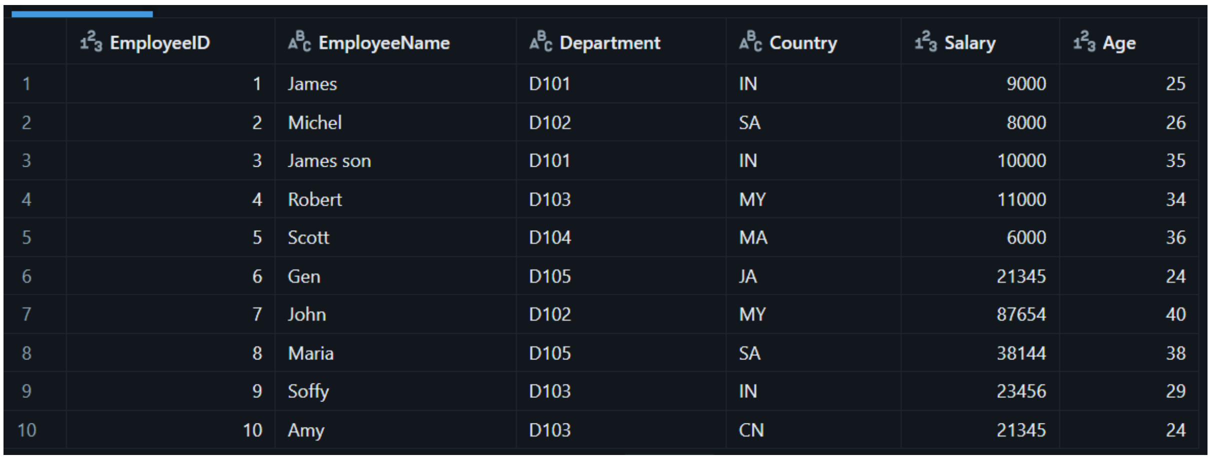
```python
country_df.display()
department_df.display()
employee_df.display()
```



	DepartmentID	DepartmentName
1	D101	Sales
2	D102	Marketing
3	D103	Finance
4	D104	Support
5	D105	HR

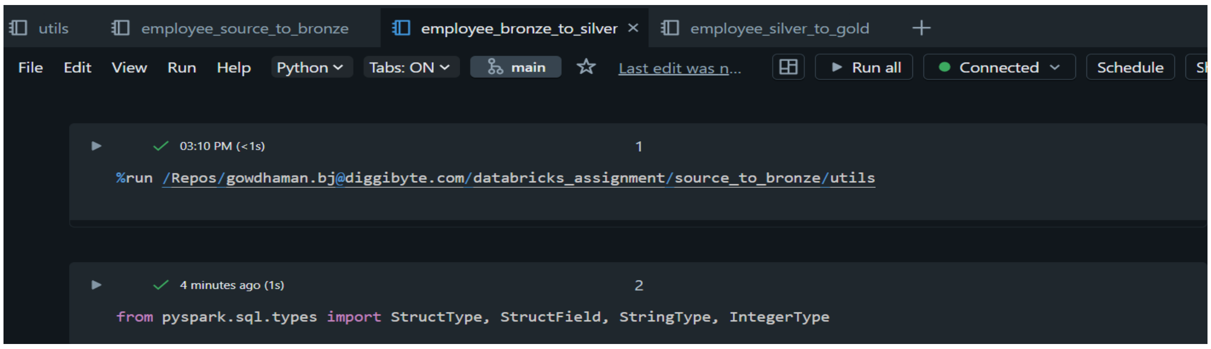


	CountryCode	CountryName
1	CN	China
2	IN	India
3	SA	South Africa
4	JA	Japan
5	MY	Malaysia
6	MA	Morocco



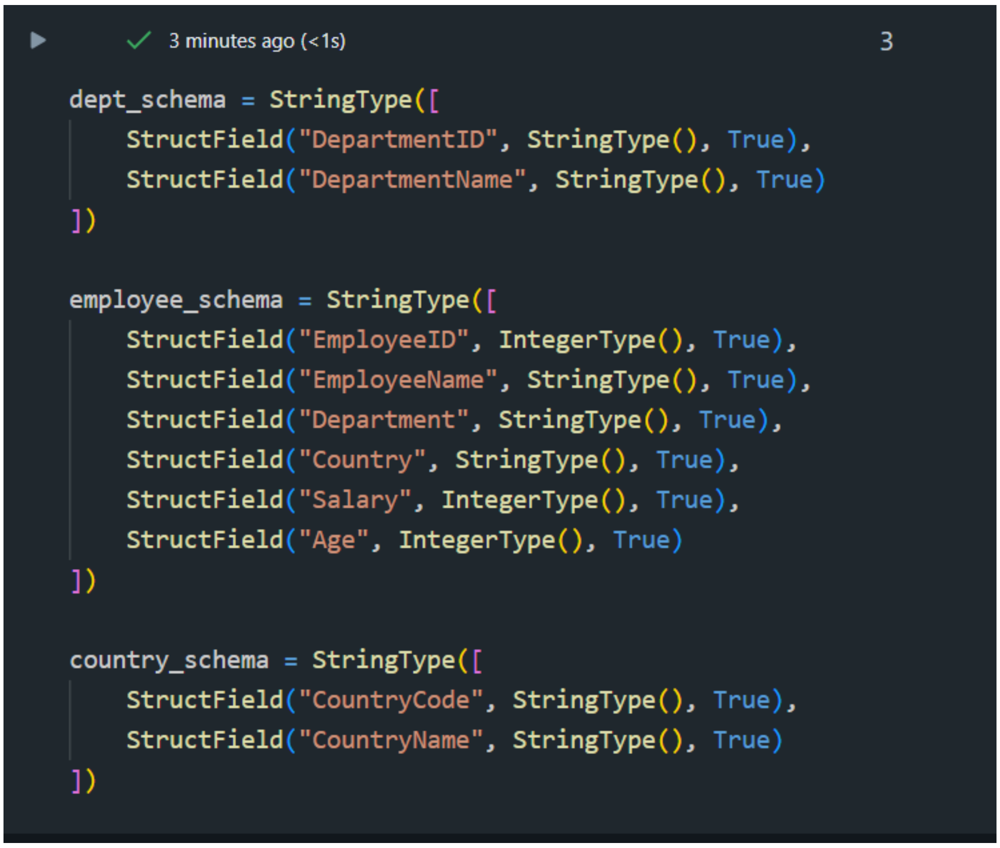
	EmployeeID	EmployeeName	Department	Country	Salary	Age
1	1	James	D101	IN	9000	25
2	2	Michel	D102	SA	8000	26
3	3	James son	D101	IN	10000	35
4	4	Robert	D103	MY	11000	34
5	5	Scott	D104	MA	6000	36
6	6	Gen	D105	JA	21345	24
7	7	John	D102	MY	87654	40
8	8	Maria	D105	SA	38144	38
9	9	Soffy	D103	IN	23456	29
10	10	Amy	D103	CN	21345	24

4. In **employee_bronze_to_silver**, call utils notebook in this notebook.
   Read the file located in DBFS location source_to_bronze with as data frame different read methods using custom schema.
   (dbfs access is not their)

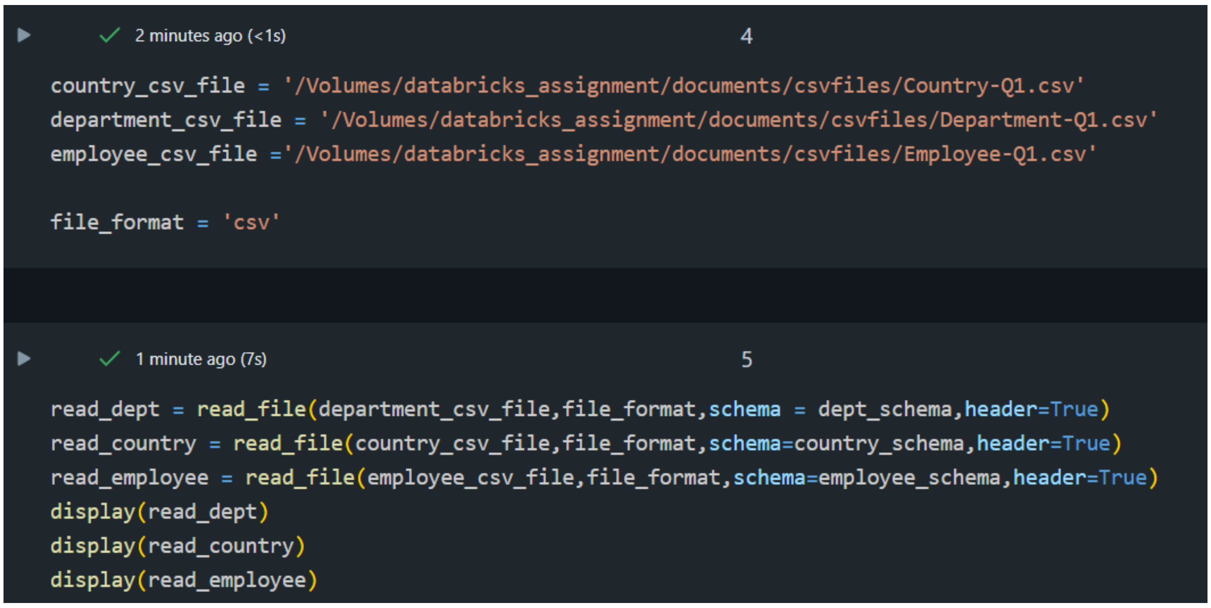


```
%run /Repos/gowdhaman.bj@diggibyte.com/databricks_assignment/source_to_bronze/utils
```

```
from pyspark.sql.types import StructType, StructField, StringType, IntegerType
```


3 minutes ago (<1s) 3

```
dept_schema = StringType([
    StructField("DepartmentID", StringType(), True),
    StructField("DepartmentName", StringType(), True)
])

employee_schema = StringType([
    StructField("EmployeeID", IntegerType(), True),
    StructField("EmployeeName", StringType(), True),
    StructField("Department", StringType(), True),
    StructField("Country", StringType(), True),
    StructField("Salary", IntegerType(), True),
    StructField("Age", IntegerType(), True)
])

country_schema = StringType([
    StructField("CountryCode", StringType(), True),
    StructField("CountryName", StringType(), True)
])
```

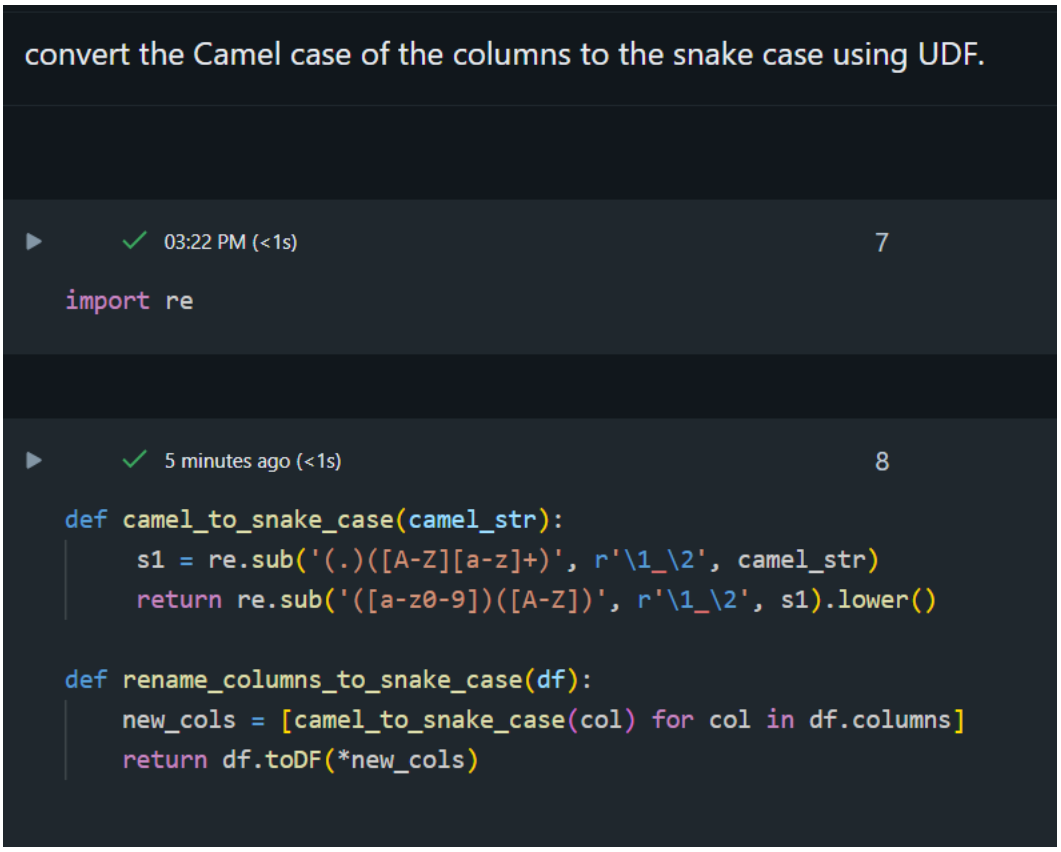

2 minutes ago (<1s) 4

```
country_csv_file = '/Volumes/databricks_assignment/documents/csvfiles/Country-Q1.csv'
department_csv_file = '/Volumes/databricks_assignment/documents/csvfiles/Department-Q1.csv'
employee_csv_file ='/Volumes/databricks_assignment/documents/csvfiles/Employee-Q1.csv'

file_format = 'csv'
```

1 minute ago (7s) 5

```
read_dept = read_file(department_csv_file,file_format,schema = dept_schema,header=True)
read_country = read_file(country_csv_file,file_format,schema=country_schema,header=True)
read_employee = read_file(employee_csv_file,file_format,schema=employee_schema,header=True)
display(read_dept)
display(read_country)
display(read_employee)
```

5. convert the Camel case of the columns to the snake case using UDF.



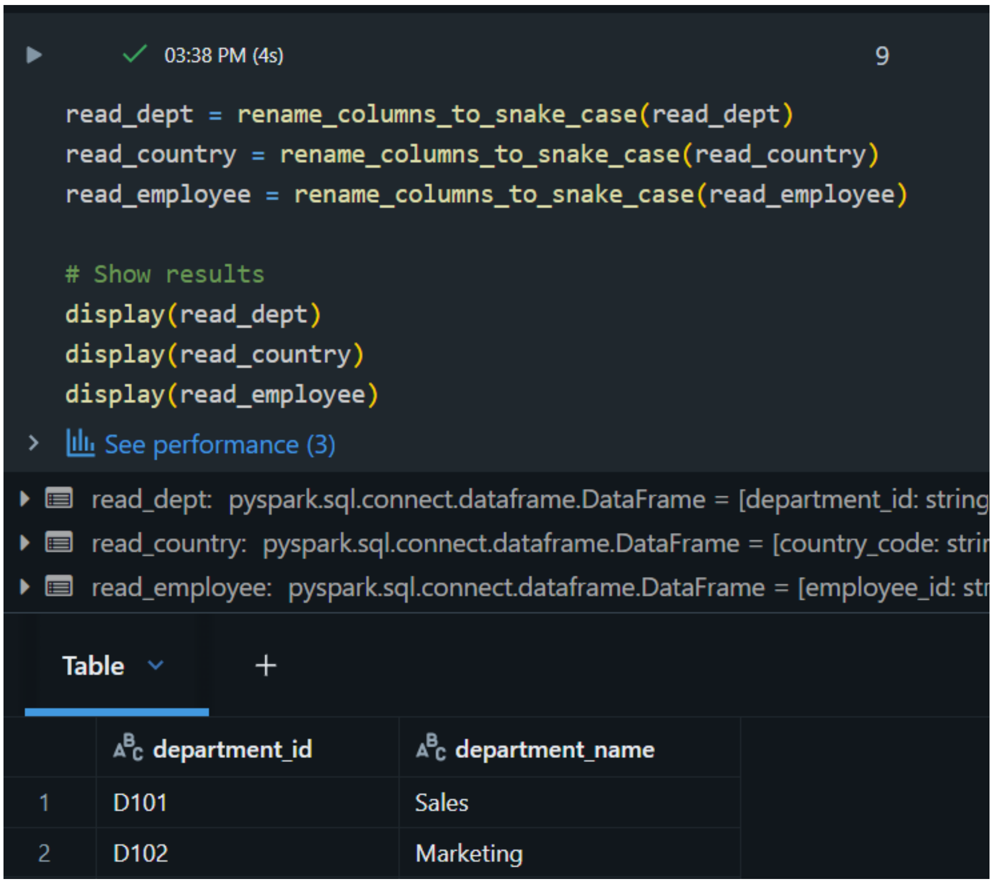
convert the Camel case of the columns to the snake case using UDF.

03:22 PM (<1s) 7

```
import re
```

5 minutes ago (<1s) 8

```
def camel_to_snake_case(camel_str):
     s1 = re.sub('(.)([A-Z][a-z]+)', r'\1_\2', camel_str)
     return re.sub('([a-z0-9])([A-Z])', r'\1_\2', s1).lower()

def rename_columns_to_snake_case(df):
    new_cols = [camel_to_snake_case(col) for col in df.columns]
    return df.toDF(*new_cols)
```



03:38 PM (4s) 9

```
read_dept = rename_columns_to_snake_case(read_dept)
read_country = rename_columns_to_snake_case(read_country)
read_employee = rename_columns_to_snake_case(read_employee)

# Show results
display(read_dept)
display(read_country)
display(read_employee)
```

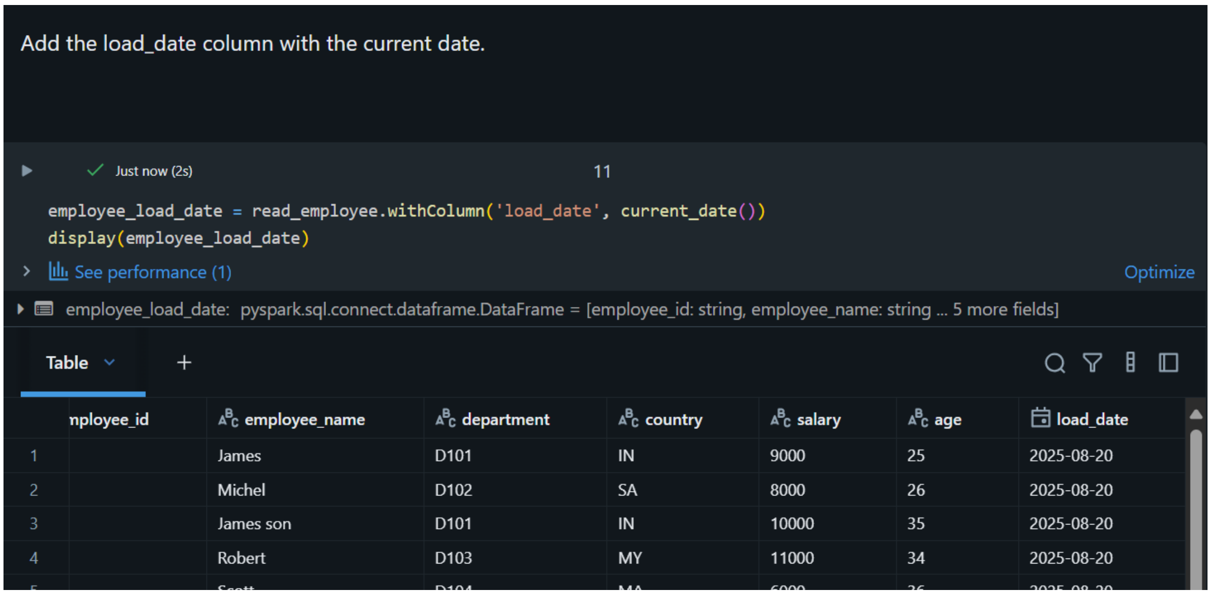
See performance (3)

```
read_dept:  pyspark.sql.connect.dataframe.DataFrame = [department_id: string
read_country:  pyspark.sql.connect.dataframe.DataFrame = [country_code: strir
read_employee:  pyspark.sql.connect.dataframe.DataFrame = [employee_id: str
```

Table

	department_id	department_name
1	D101	Sales
2	D102	Marketing

6. Add the **load_date** column with the current date.
   The primary key is EmployeeID, the Database name is Employee_info, Table name is dim_employee.
   write the DF as a delta table to the location /silver/db_name/table_name.



Add the load_date column with the current date.

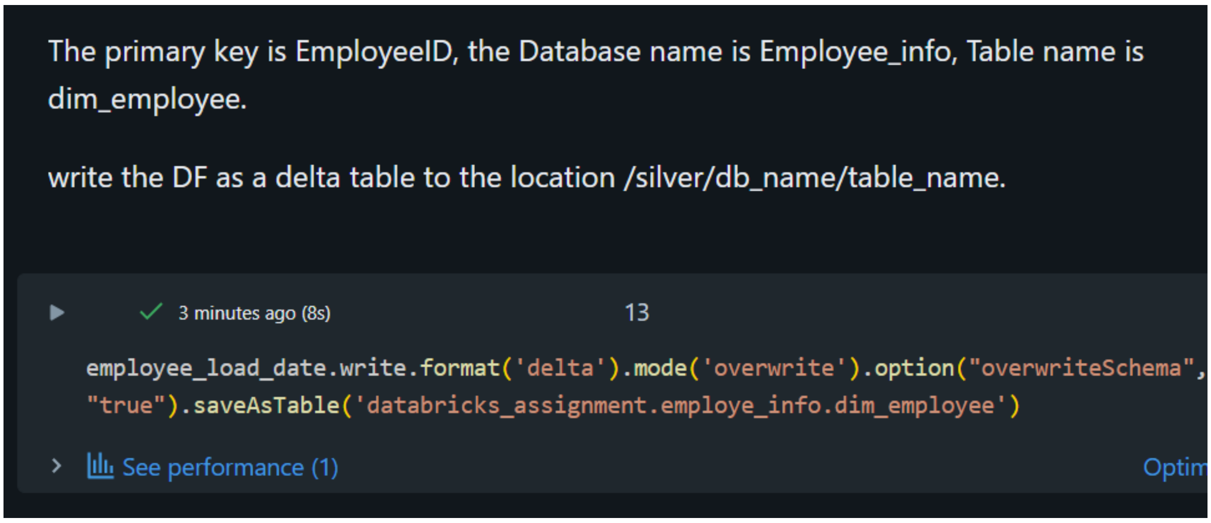
Just now (2s) 11

```
employee_load_date = read_employee.withColumn('load_date', current_date())
display(employee_load_date)
```

See performance (1) Optimize

employee_load_date: pyspark.sql.connect.dataframe.DataFrame = [employee_id: string, employee_name: string ... 5 more fields]

	nployee_id	employee_name	department	country	salary	age	load_date
1		James	D101	IN	9000	25	2025-08-20
2		Michel	D102	SA	8000	26	2025-08-20
3		James son	D101	IN	10000	35	2025-08-20
4		Robert	D103	MY	11000	34	2025-08-20



The primary key is EmployeeID, the Database name is Employee_info, Table name is dim_employee.

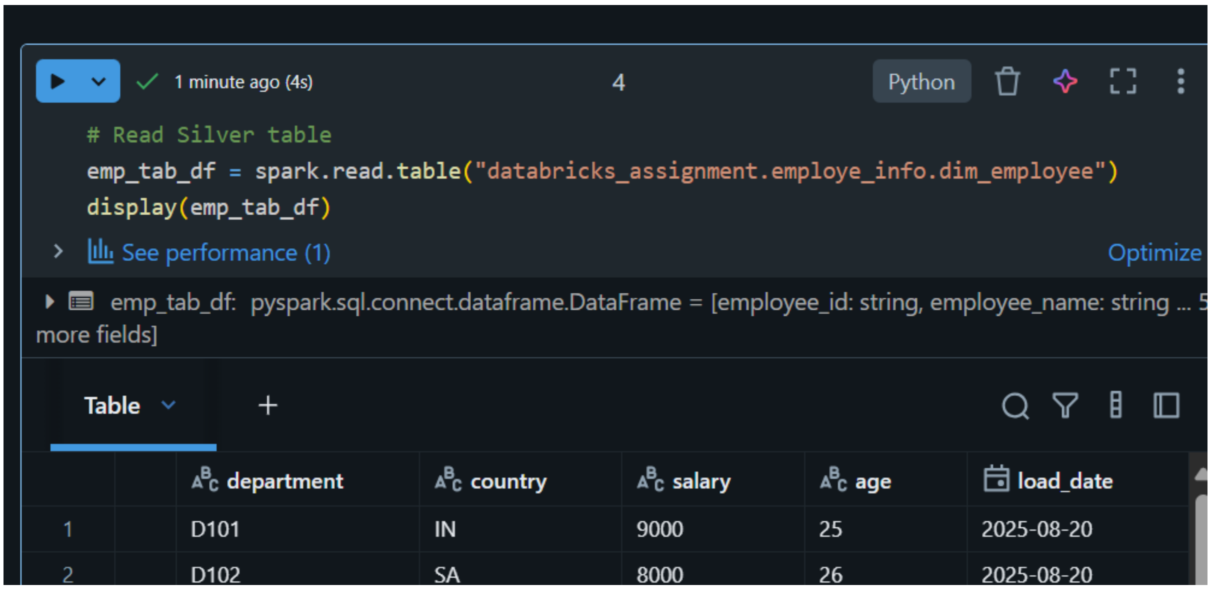
write the DF as a delta table to the location /silver/db_name/table_name.

3 minutes ago (8s) 13

```
employee_load_date.write.format('delta').mode('overwrite').option("overwriteSchema",
"true").saveAsTable('databricks_assignment.employe_info.dim_employee')
```

See performance (1) Optim

7. In gold notebook employee_silver_to_gold, call utils notebook in this notebook Read the table stored in a silver layer as DataFrame and select the columns based on the following requirements.



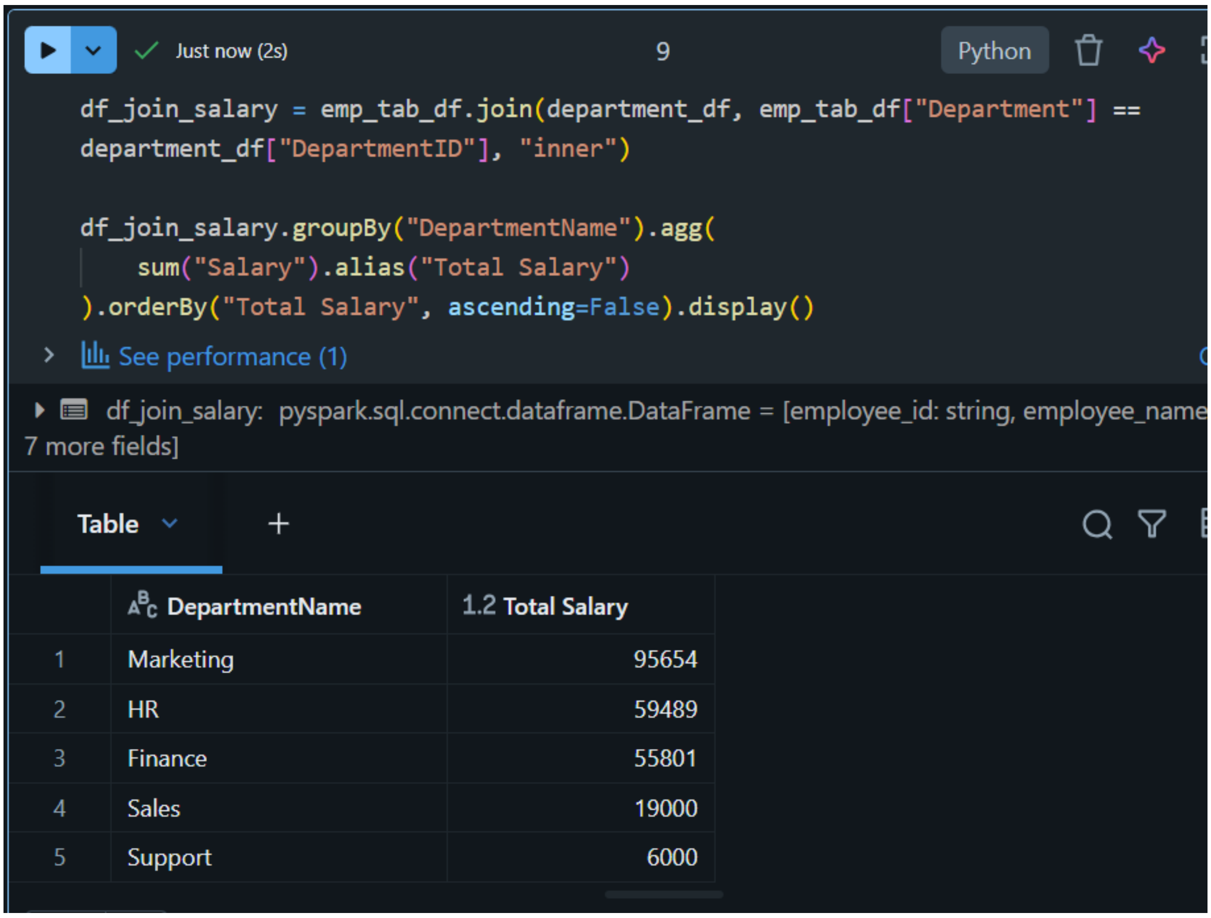
```
# Read Silver table
emp_tab_df = spark.read.table("databricks_assignment.employe_info.dim_employee")
display(emp_tab_df)
```

emp_tab_df: pyspark.sql.connect.dataframe.DataFrame = [employee_id: string, employee_name: string ... 5 more fields]

	department	country	salary	age	load_date
1	D101	IN	9000	25	2025-08-20
2	D102	SA	8000	26	2025-08-20

8. Requirements:

- Find the salary of each department in descending order.

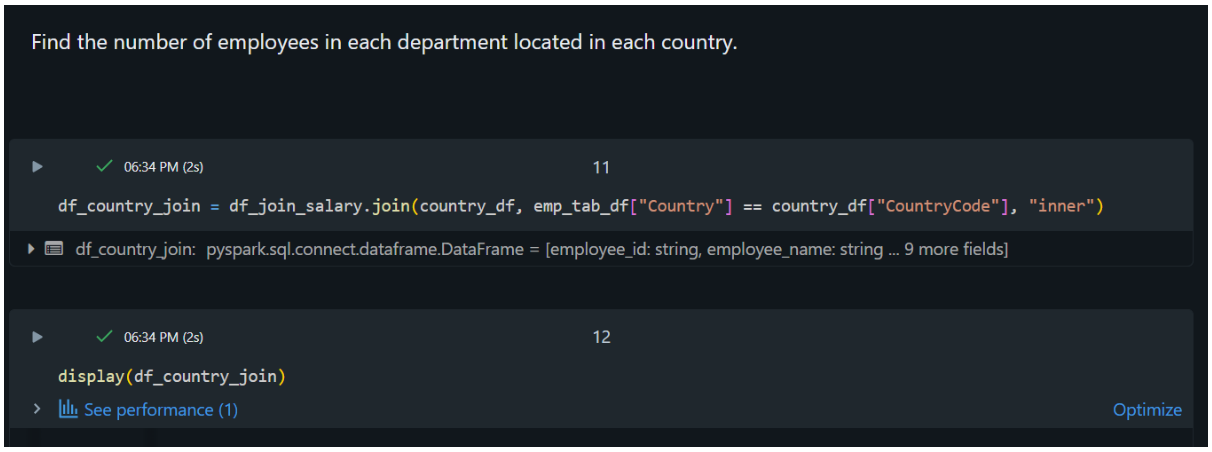


```
df_join_salary = emp_tab_df.join(department_df, emp_tab_df["Department"] ==
department_df["DepartmentID"], "inner")

df_join_salary.groupBy("DepartmentName").agg(
    sum("Salary").alias("Total Salary")
).orderBy("Total Salary", ascending=False).display()
```

df_join_salary: pyspark.sql.connect.dataframe.DataFrame = [employee_id: string, employee_name
7 more fields]

	DepartmentName	Total Salary
1	Marketing	95654
2	HR	59489
3	Finance	55801
4	Sales	19000
5	Support	6000

Find the number of employees in each department located in each country.

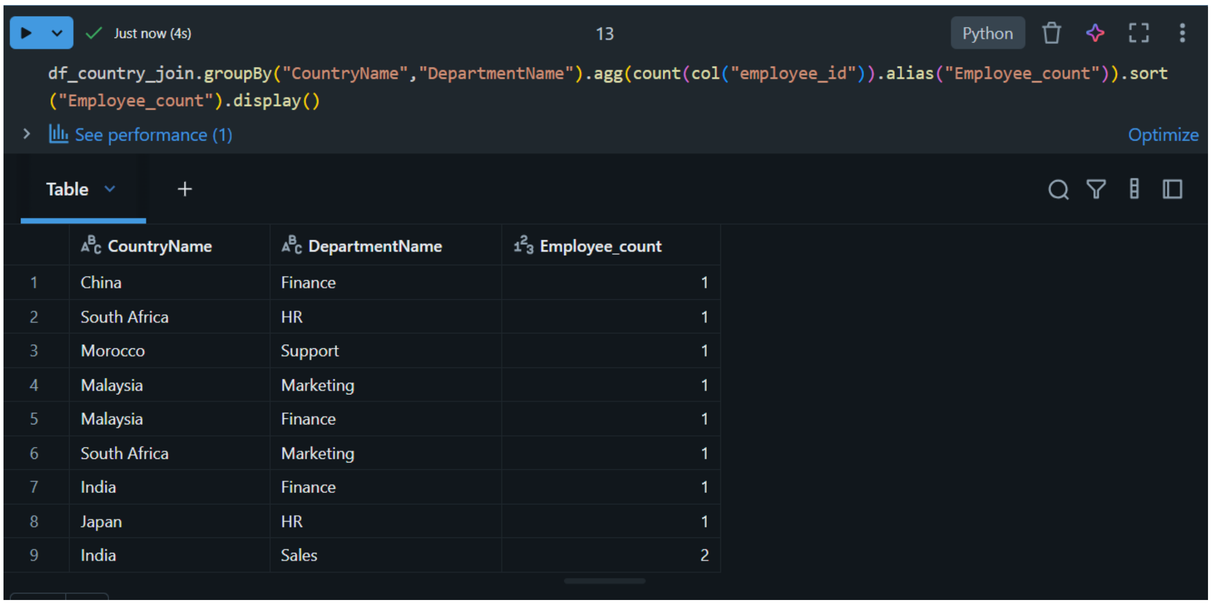


Find the number of employees in each department located in each country.

```
df_country_join = df_join_salary.join(country_df, emp_tab_df["Country"] == country_df["CountryCode"], "inner")
```

```
df_country_join:  pyspark.sql.connect.dataframe.DataFrame = [employee_id: string, employee_name: string ... 9 more fields]
```

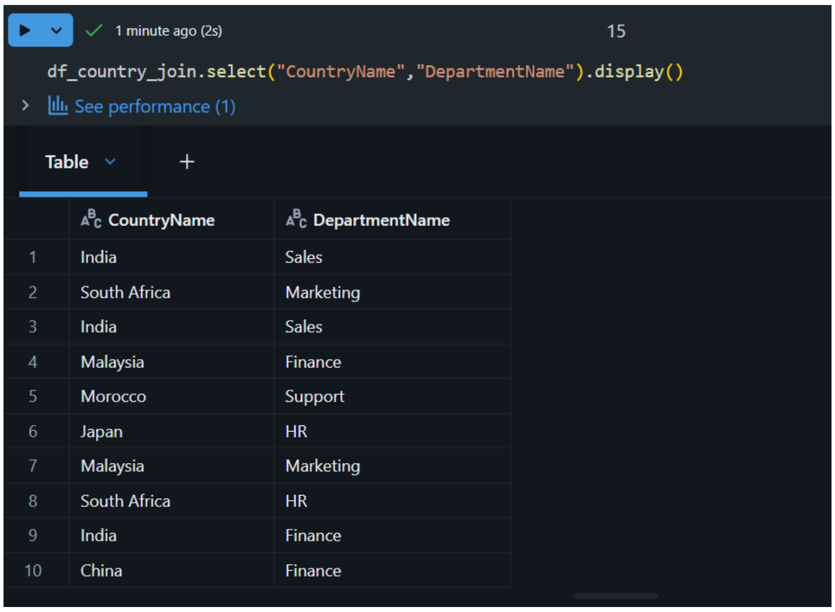
```
display(df_country_join)
```



```
df_country_join.groupBy("CountryName","DepartmentName").agg(count(col("employee_id")).alias("Employee_count")).sort
("Employee_count").display()
```

	CountryName	DepartmentName	Employee_count
1	China	Finance	1
2	South Africa	HR	1
3	Morocco	Support	1
4	Malaysia	Marketing	1
5	Malaysia	Finance	1
6	South Africa	Marketing	1
7	India	Finance	1
8	Japan	HR	1
9	India	Sales	2

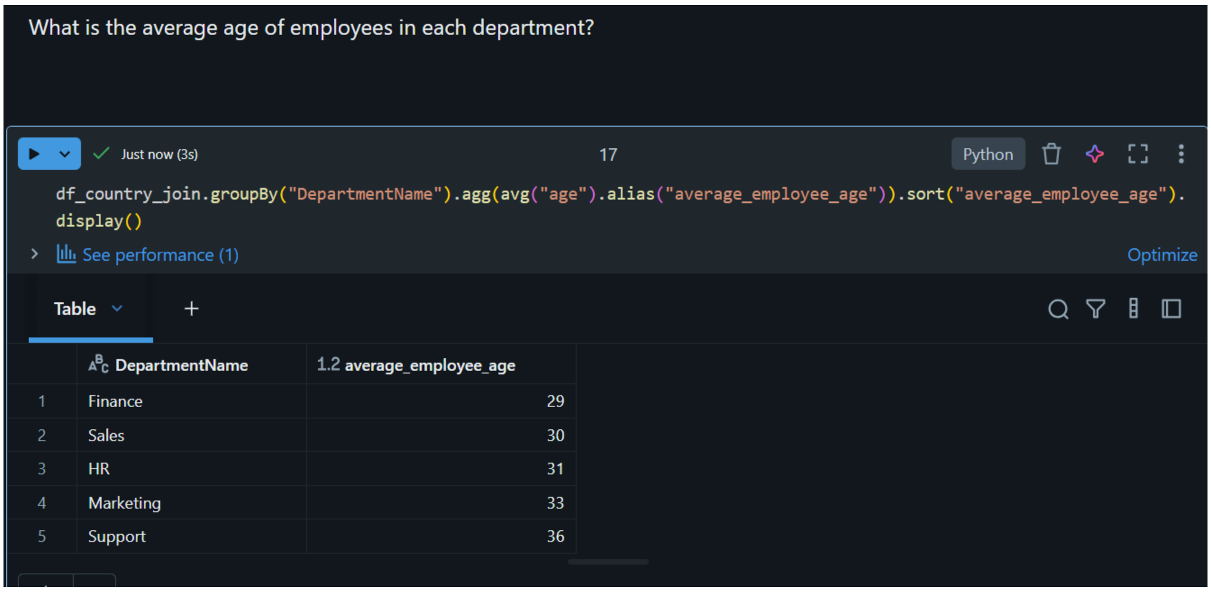
List the department names along with their corresponding country names



```
df_country_join.select("CountryName","DepartmentName").display()
```

	CountryName	DepartmentName
1	India	Sales
2	South Africa	Marketing
3	India	Sales
4	Malaysia	Finance
5	Morocco	Support
6	Japan	HR
7	Malaysia	Marketing
8	South Africa	HR
9	India	Finance
10	China	Finance

What is the average age of employees in each department?



What is the average age of employees in each department?

Just now (3s) 17 Python

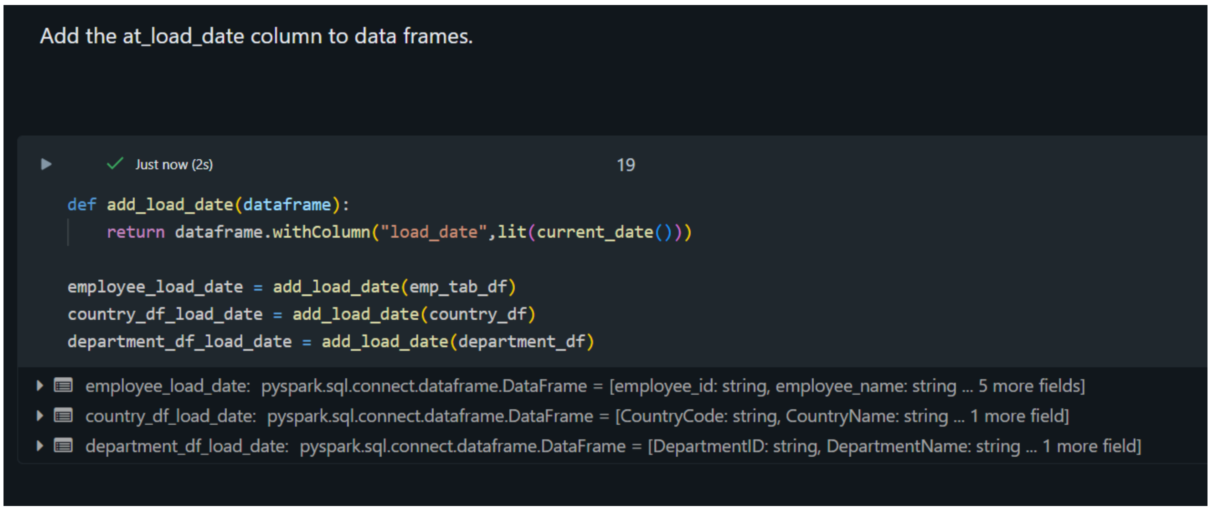
```
df_country_join.groupBy("DepartmentName").agg(avg("age").alias("average_employee_age")).sort("average_employee_age").
display()
```

See performance (1) Optimize

Table

	DepartmentName	1.2 average_employee_age
1	Finance	29
2	Sales	30
3	HR	31
4	Marketing	33
5	Support	36

Add the at_load_date column to data frames.

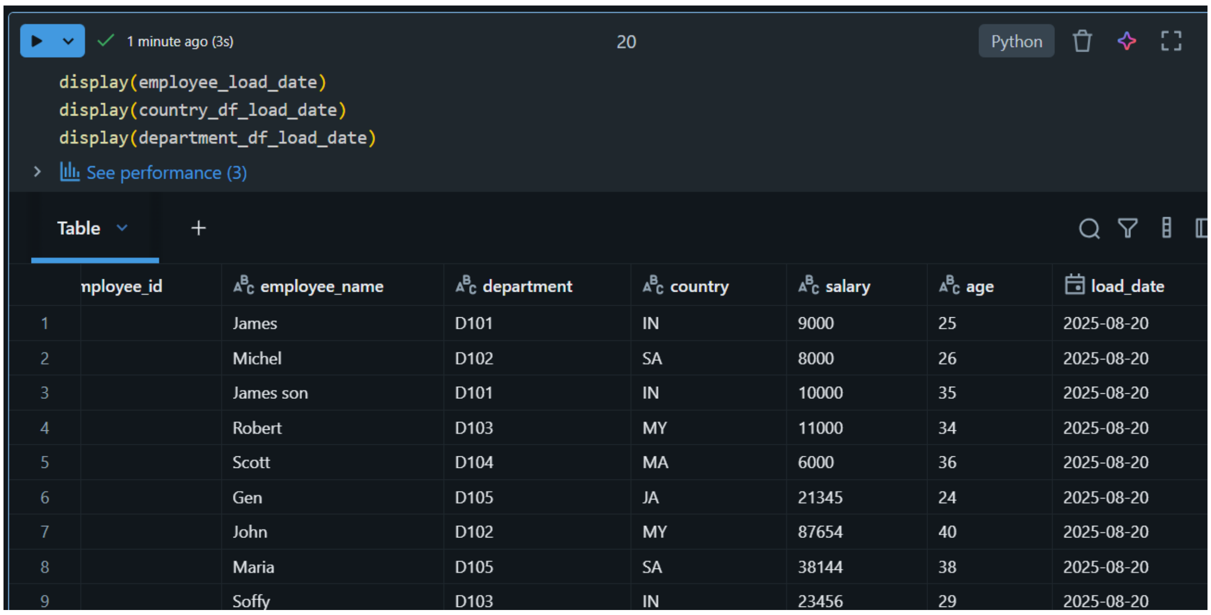


Add the at_load_date column to data frames.

Just now (2s) 19

```
def add_load_date(dataframe):
    return dataframe.withColumn("load_date",lit(current_date()))

employee_load_date = add_load_date(emp_tab_df)
country_df_load_date = add_load_date(country_df)
department_df_load_date = add_load_date(department_df)
```

```
employee_load_date:  pyspark.sql.connect.dataframe.DataFrame = [employee_id: string, employee_name: string ... 5 more fields]
country_df_load_date:  pyspark.sql.connect.dataframe.DataFrame = [CountryCode: string, CountryName: string ... 1 more field]
department_df_load_date:  pyspark.sql.connect.dataframe.DataFrame = [DepartmentID: string, DepartmentName: string ... 1 more field]
```



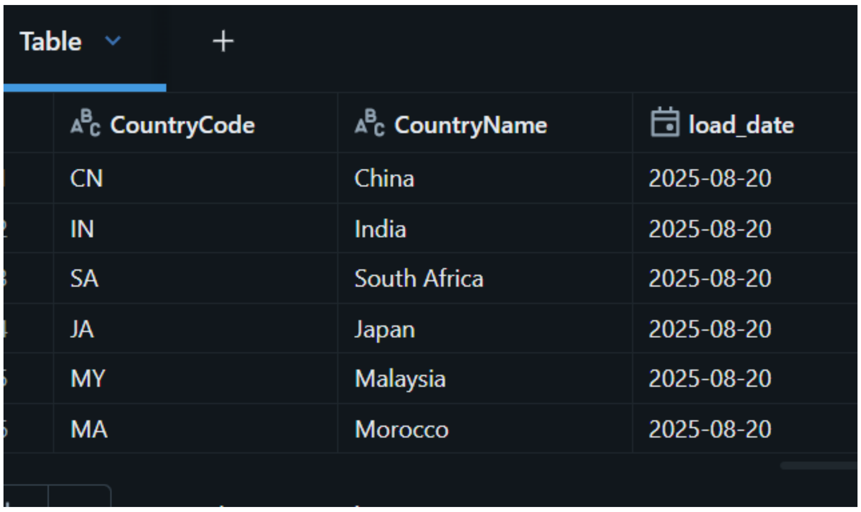
1 minute ago (3s) 20 Python

```
display(employee_load_date)
display(country_df_load_date)
display(department_df_load_date)
```

See performance (3)

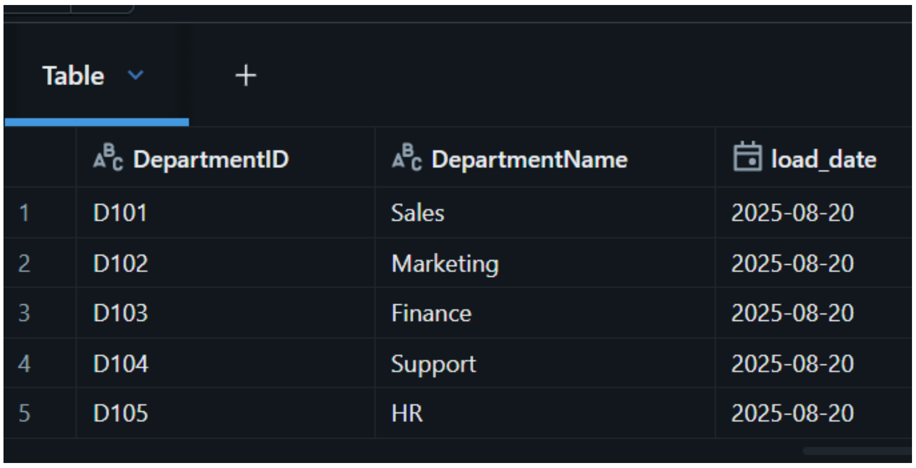
Table

	nployee_id	employee_name	department	country	salary	age	load_date
1		James	D101	IN	9000	25	2025-08-20
2		Michel	D102	SA	8000	26	2025-08-20
3		James son	D101	IN	10000	35	2025-08-20
4		Robert	D103	MY	11000	34	2025-08-20
5		Scott	D104	MA	6000	36	2025-08-20
6		Gen	D105	JA	21345	24	2025-08-20
7		John	D102	MY	87654	40	2025-08-20
8		Maria	D105	SA	38144	38	2025-08-20
9		Soffy	D103	IN	23456	29	2025-08-20



Table

	CountryCode	CountryName	load_date
	CN	China	2025-08-20
	IN	India	2025-08-20
	SA	South Africa	2025-08-20
	JA	Japan	2025-08-20
	MY	Malaysia	2025-08-20
	MA	Morocco	2025-08-20



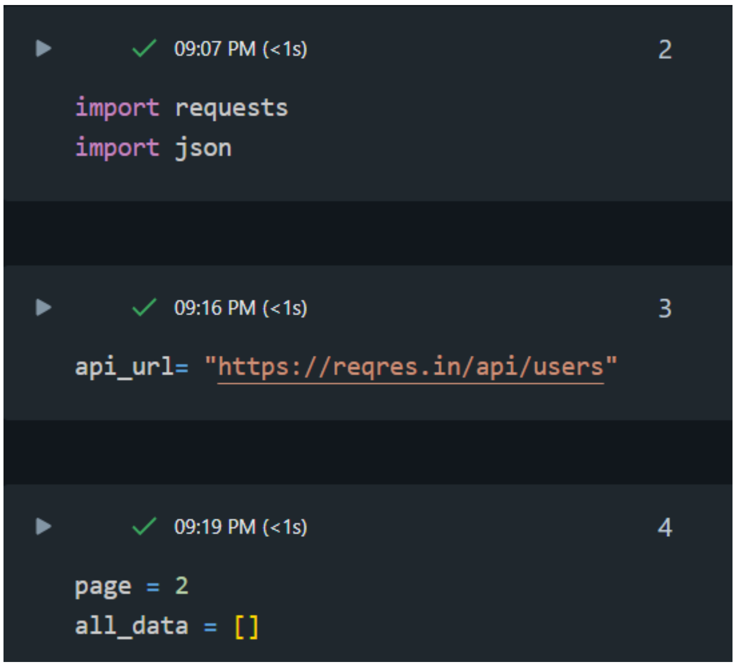
Table

	DepartmentID	DepartmentName	load_date
1	D101	Sales	2025-08-20
2	D102	Marketing	2025-08-20
3	D103	Finance	2025-08-20
4	D104	Support	2025-08-20
5	D105	HR	2025-08-20

Write the df to dbfs location /gold/employee/table_name(fact_employee) with overwrite and replace where condition on at_load_date.

Question: 2

Api: https://reqres.in/api/users?page=2


09:07 PM (<1s) 2

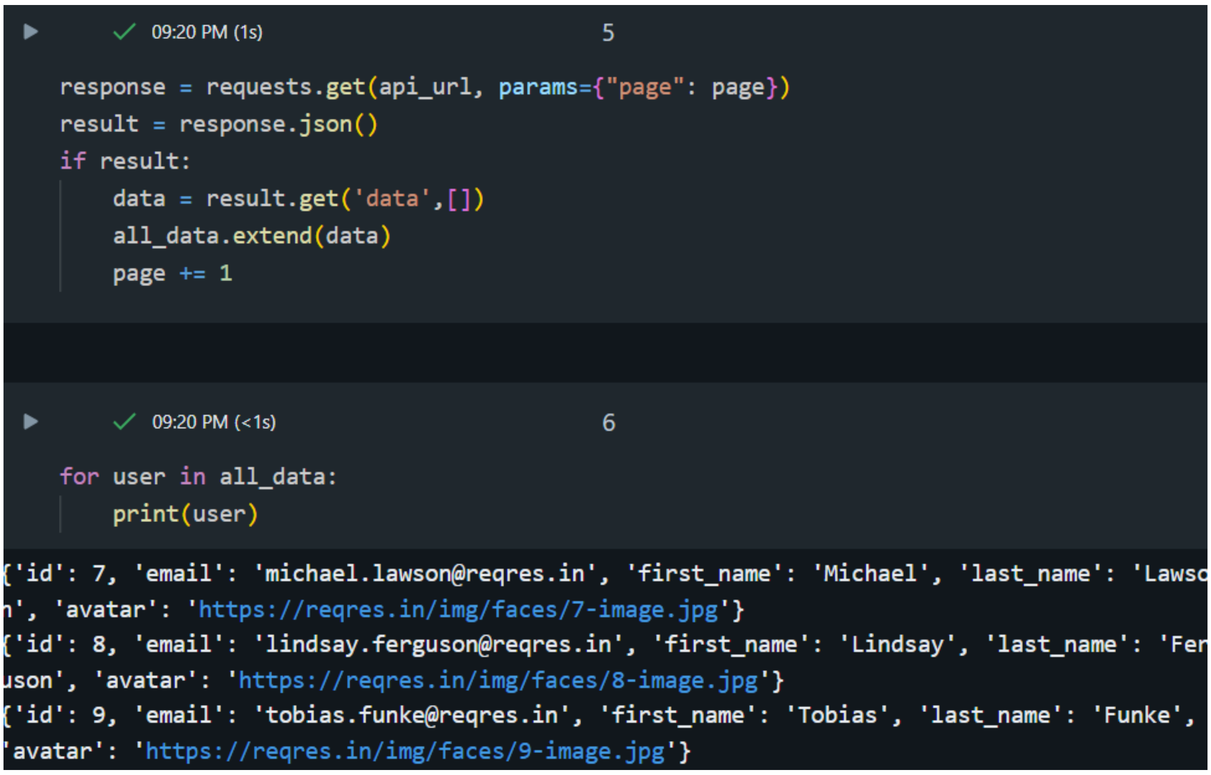
```python
import requests
import json
```

09:16 PM (<1s) 3

```python
api_url= "https://reqres.in/api/users"
```

09:19 PM (<1s) 4

```python
page = 2
all_data = []
```


09:20 PM (1s) 5

```python
response = requests.get(api_url, params={"page": page})
result = response.json()
if result:
    data = result.get('data',[])
    all_data.extend(data)
    page += 1
```

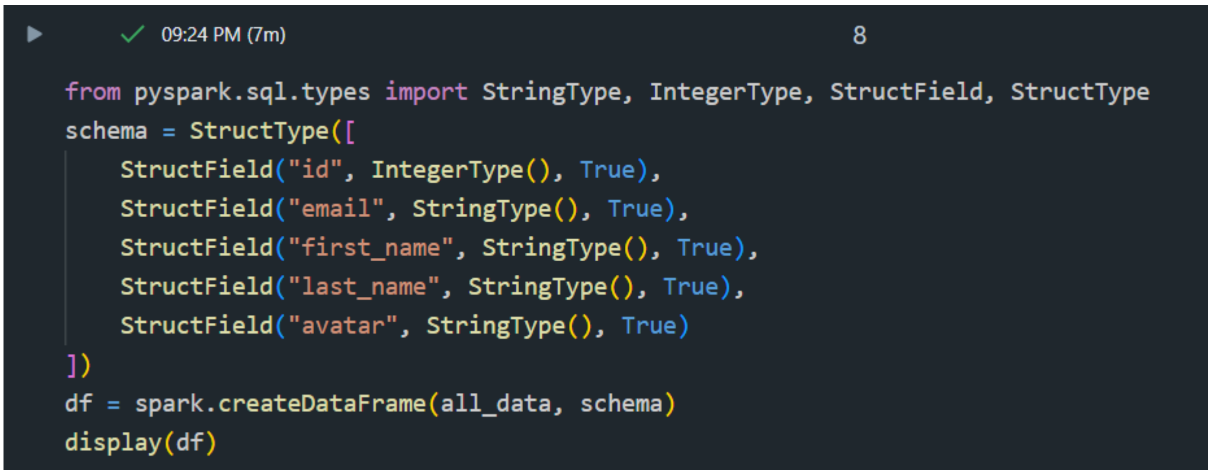
09:20 PM (<1s) 6

```python
for user in all_data:
    print(user)
```

```
{'id': 7, 'email': 'michael.lawson@reqres.in', 'first_name': 'Michael', 'last_name': 'Lawso
n', 'avatar': 'https://reqres.in/img/faces/7-image.jpg'}
{'id': 8, 'email': 'lindsay.ferguson@reqres.in', 'first_name': 'Lindsay', 'last_name': 'Fer
uson', 'avatar': 'https://reqres.in/img/faces/8-image.jpg'}
{'id': 9, 'email': 'tobias.funke@reqres.in', 'first_name': 'Tobias', 'last_name': 'Funke',
'avatar': 'https://reqres.in/img/faces/9-image.jpg'}
```

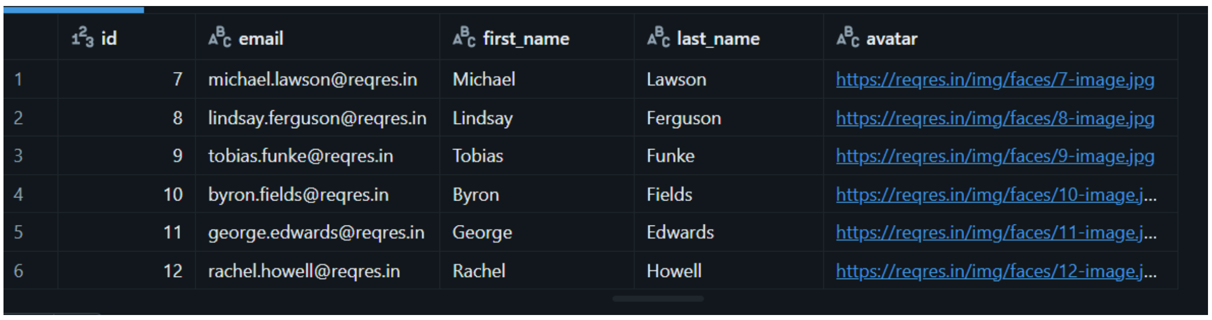
Read the data frame with a custom schema

Flatten the dataframe



09:24 PM (7m) 8

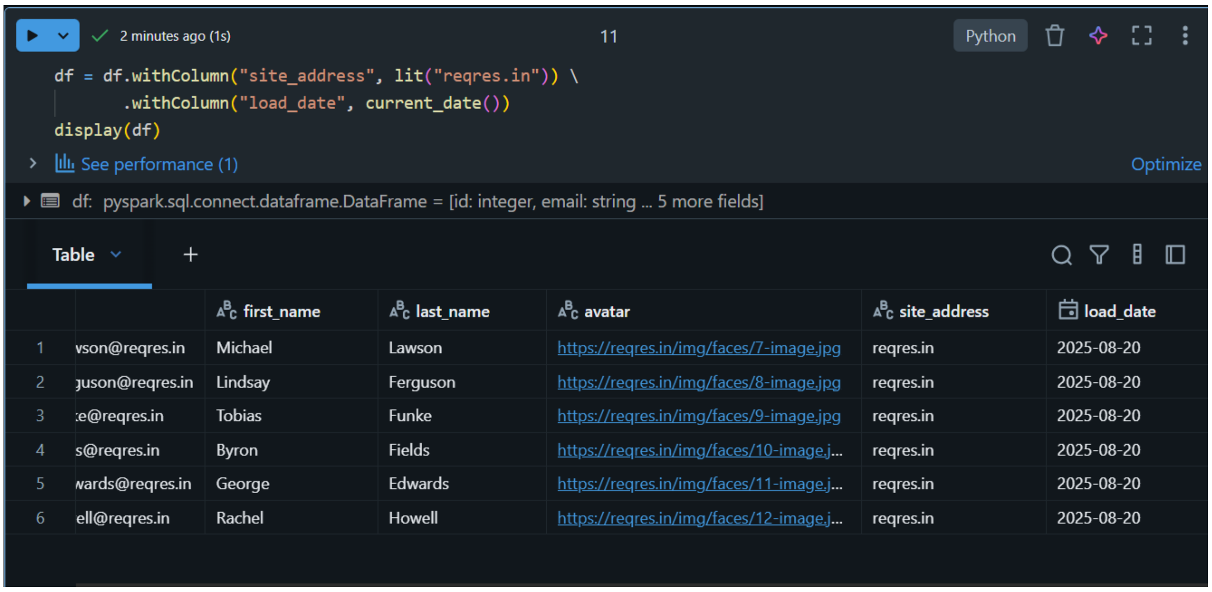
```
from pyspark.sql.types import StringType, IntegerType, StructField, StructType
schema = StructType([
    StructField("id", IntegerType(), True),
    StructField("email", StringType(), True),
    StructField("first_name", StringType(), True),
    StructField("last_name", StringType(), True),
    StructField("avatar", StringType(), True)
])
df = spark.createDataFrame(all_data, schema)
display(df)
```



	id	email	first_name	last_name	avatar
1	7	michael.lawson@reqres.in	Michael	Lawson	https://reqres.in/img/faces/7-image.jpg
2	8	lindsay.ferguson@reqres.in	Lindsay	Ferguson	https://reqres.in/img/faces/8-image.jpg
3	9	tobias.funke@reqres.in	Tobias	Funke	https://reqres.in/img/faces/9-image.jpg
4	10	byron.fields@reqres.in	Byron	Fields	https://reqres.in/img/faces/10-image.j...
5	11	george.edwards@reqres.in	George	Edwards	https://reqres.in/img/faces/11-image.j...
6	12	rachel.howell@reqres.in	Rachel	Howell	https://reqres.in/img/faces/12-image.j...

Derive a new column from email as site_address with values(reqres.in)

Add load_date with the current date.



2 minutes ago (1s) 11

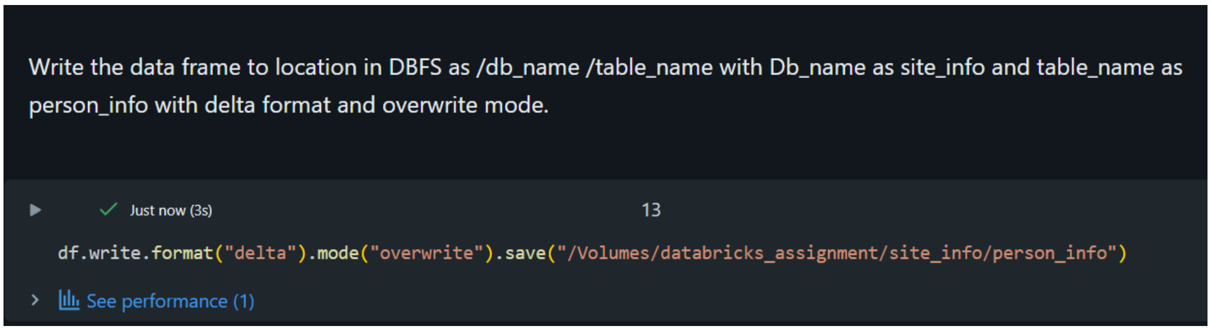
```
df = df.withColumn("site_address", lit("reqres.in")) \
       .withColumn("load_date", current_date())
display(df)
```

See performance (1)

df: pyspark.sql.connect.dataframe.DataFrame = [id: integer, email: string ... 5 more fields]

		first_name	last_name	avatar	site_address	load_date
1	vson@reqres.in	Michael	Lawson	https://reqres.in/img/faces/7-image.jpg	reqres.in	2025-08-20
2	juson@reqres.in	Lindsay	Ferguson	https://reqres.in/img/faces/8-image.jpg	reqres.in	2025-08-20
3	:e@reqres.in	Tobias	Funke	https://reqres.in/img/faces/9-image.jpg	reqres.in	2025-08-20
4	s@reqres.in	Byron	Fields	https://reqres.in/img/faces/10-image.j...	reqres.in	2025-08-20
5	vards@reqres.in	George	Edwards	https://reqres.in/img/faces/11-image.j...	reqres.in	2025-08-20
6	ell@reqres.in	Rachel	Howell	https://reqres.in/img/faces/12-image.j...	reqres.in	2025-08-20

Write the data frame to location in DBFS as /db_name /table_name with Db_name as site_info and table_name as person_info with delta format and overwrite mode.

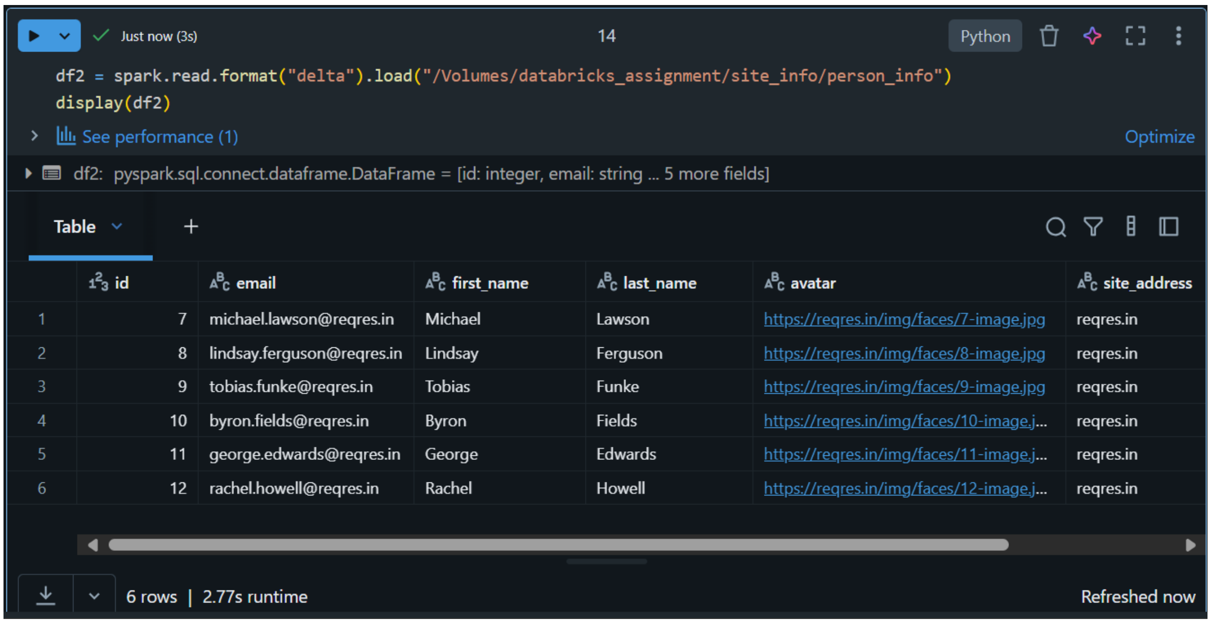


Write the data frame to location in DBFS as /db_name /table_name with Db_name as site_info and table_name as person_info with delta format and overwrite mode.

Just now (3s) 13

```
df.write.format("delta").mode("overwrite").save("/Volumes/databricks_assignment/site_info/person_info")
```

See performance (1)



Just now (3s) 14

```
df2 = spark.read.format("delta").load("/Volumes/databricks_assignment/site_info/person_info")
display(df2)
```

See performance (1)

df2: pyspark.sql.connect.dataframe.DataFrame = [id: integer, email: string ... 5 more fields]

	id	email	first_name	last_name	avatar	site_address
1	7	michael.lawson@reqres.in	Michael	Lawson	https://reqres.in/img/faces/7-image.jpg	reqres.in
2	8	lindsay.ferguson@reqres.in	Lindsay	Ferguson	https://reqres.in/img/faces/8-image.jpg	reqres.in
3	9	tobias.funke@reqres.in	Tobias	Funke	https://reqres.in/img/faces/9-image.jpg	reqres.in
4	10	byron.fields@reqres.in	Byron	Fields	https://reqres.in/img/faces/10-image.j...	reqres.in
5	11	george.edwards@reqres.in	George	Edwards	https://reqres.in/img/faces/11-image.j...	reqres.in
6	12	rachel.howell@reqres.in	Rachel	Howell	https://reqres.in/img/faces/12-image.j...	reqres.in

6 rows | 2.77s runtime

Refreshed now