

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 ☆

Send feedback

Share

Create ▾

Q Search

Type ▾

Owner ▾

Last modified ▾

Name <div>≡↑</div>	Type	Owner	Created at	<div></div>
<div></div> employee_source_to_bronze	Notebook	gowdhaman.bj@...	Aug 20, 2025, 01:...	
<div></div> utils	Notebook	gowdhaman.bj@...	Aug 20, 2025, 01:...	

- 1 Notebook in bronze to silver as employee_bronze_to_silver

bronze_to_silver ☆

Send feedback

Share

Create ▾

Q Search

Type ▾

Owner ▾

Last modified ▾

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 ☆

Send feedback

⋮

Share

Create ▾

Q Search

Type ▾

Owner ▾

Last modified ▾

Name ↕

Type

Owner

Created at

☰

📓 employee_silver_to_gold

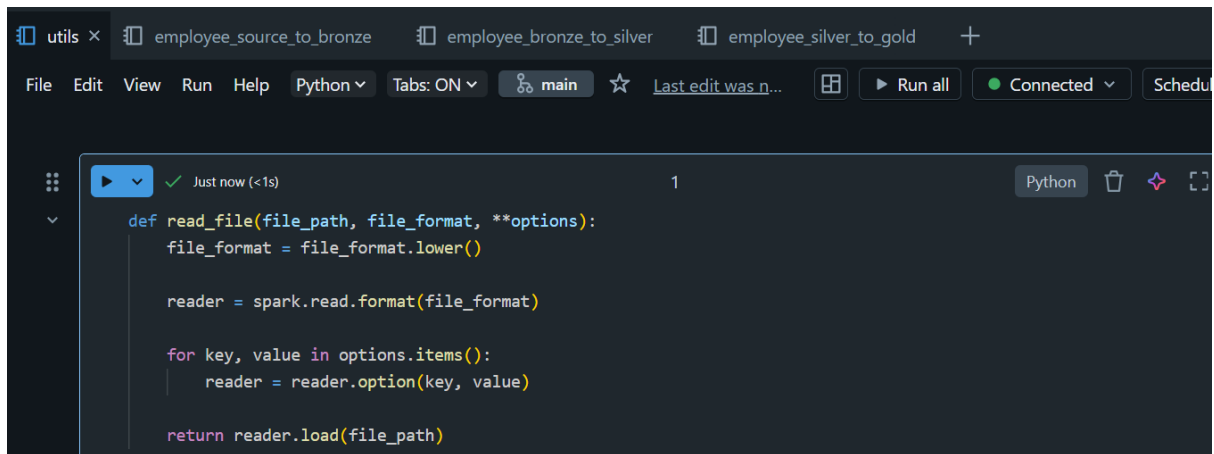
Notebook

gowdhaman.bj@...

Aug 20, 2025, 01:...

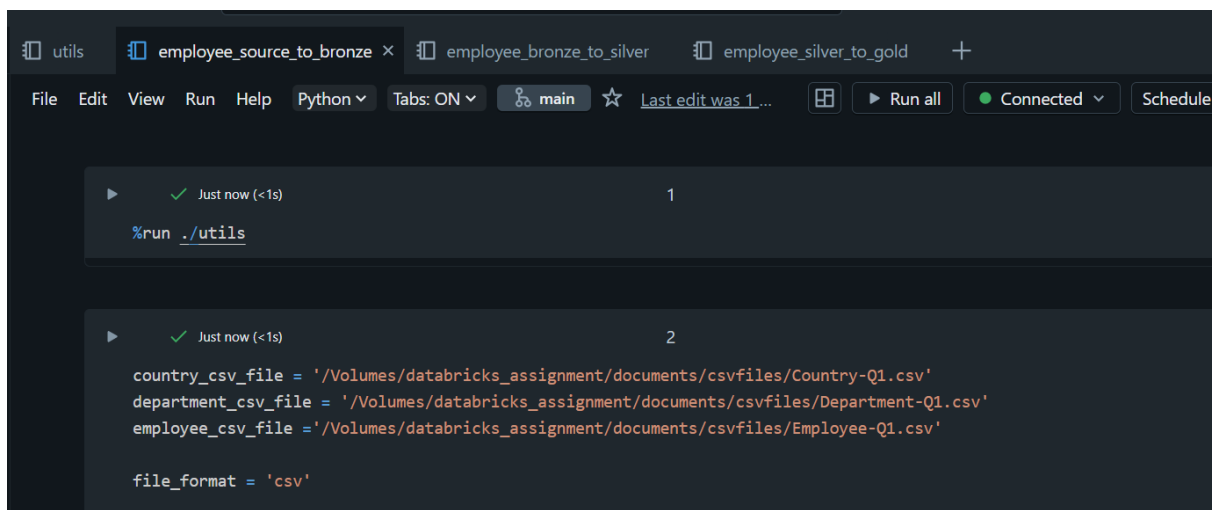
3. Read the 3 datasets as Dataframe in **employee_source_to_bronze**, call utils notebook in this notebook, and write to a location in DBFS, as

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




The screenshot shows a Databricks IDE interface with a tab titled 'employee_source_to_bronze'. The code editor displays a Python function named `read_file` that takes `file_path`, `file_format`, and `**options` as arguments. The function converts `file_format` to lowercase, creates a `spark.read` reader with the specified format, applies any options, and returns the loaded data as a DataFrame.

```
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)
```



The screenshot shows a Databricks IDE interface with a tab titled 'employee_source_to_bronze'. A Jupyter cell is shown with the command `%run ./utils` and a second cell containing file paths for CSV files and the file format.

```
%run ./utils  
  
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'
```



The screenshot shows a Databricks IDE interface with a tab titled 'employee_source_to_bronze'. It displays two Jupyter cells. The first cell uses the `read_file` function to load CSV files into `country_df`, `department_df`, and `employee_df`. The second cell displays these DataFrames. The output of the first cell shows the schema for each DataFrame.

```
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]

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

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

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

	I_3^2 EmployeeID	A_C^B EmployeeName	A_C^B Department	A_C^B Country	I_3^2 Salary	I_3^2 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

- 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)

```

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 ▾ Schedule S
▶ 03:10 PM (<1s) 1
%run /Repos/gowdhaman.bj@diggibyte.com/databricks_assignment/source_to_bronze/utils

▶ 4 minutes ago (1s) 2
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: string
▶ read_employee: pyspark.sql.connect.dataframe.DataFrame = [employee_id: string

Table ▼ +

	^A _C department_id	^A _C 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]

	employee_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	26	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.employee_info.dim_employee')
```

> [See performance \(1\)](#) Optimize