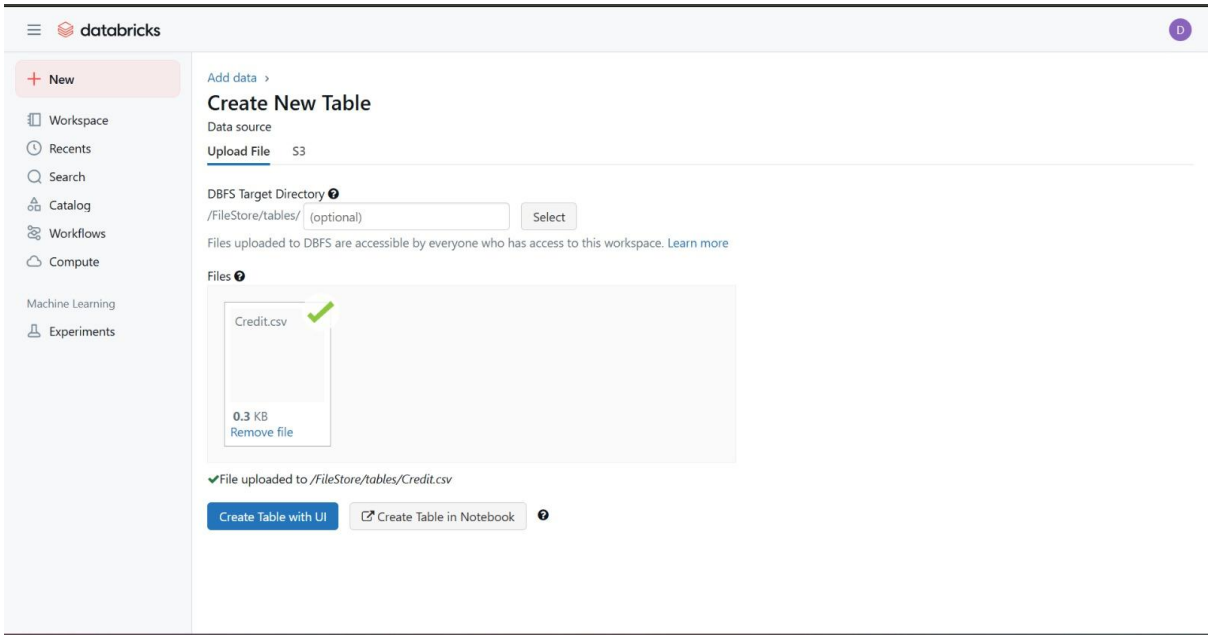
Data bricks community edition

Loading a file to Data bricks



17 hours ago (16s) 2 Python

```python
# File location and type
file_location = "/FileStore/tables/Credit.csv"
file_type = "csv"

# CSV options
infer_schema = "true"
first_row_is_header = "true"
delimiter = ","

# The applied options are for CSV files. For other file types, these will be ignored.
df = spark.read.format(file_type) \
  .option("inferSchema", infer_schema) \
  .option("header", first_row_is_header) \
  .option("sep", delimiter) \
  .load(file_location)

display(df)
```

(3) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [Credit_id: integer, Credit_Name: string ... 2 more fields]

17 hours ago (16s) 2 Python

(3) Spark Jobs

df: pyspark.sql.dataframe.DataFrame = [Credit_id: integer, Credit_Name: string ... 2 more fields]

Table

	Credit_id	Credit_Name	Credit_Type	Credit_Score
1	1	Deepthi	Master	989
2	2	Rahul	Visa	1124
3	3	Anjana	Visa	1124
4	4	Shriya	Master	765
5	1	Deepthi	Visa	678
6	2	Rahul	Master	1245
7	3	Anjana	Master	989
8	4	Shriya	Visa	765
9	1	Deepthi	Master	989
10	3	Anjana	Master	789

10 rows | 16.44s runtime Refreshed 17 hours ago

17 hours ago (<1s) 3

```python
# Create a view or table

temp_table_name = "Credit_csv"

df.createOrReplaceTempView(temp_table_name)
```

17 hours ago (1s) 4

```sql
%sql

/* Query the created temp table in a SQL cell */

select * from `Credit_csv`
```

(1) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [Credit_id: integer, Credit_Name: string ... 2 more fields]

(1) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [Credit_id: integer, Credit_Name: string ... 2 more fields]

Table

	Credit_id	Credit_Name	Credit_Type	Credit_Score
1	1	Deepthi	Master	989
2	2	Rahul	Visa	1124
3	3	Anjana	Visa	1124
4	4	Shriya	Master	765
5	1	Deepthi	Visa	678
6	2	Rahul	Master	1245
7	3	Anjana	Master	989
8	4	Shriya	Visa	765
9	1	Deepthi	Master	989
10	3	Anjana	Master	789

10 rows | 0.72s runtime

Refreshed 17 hours ago

17 hours ago (9s) 5 Python

```
# With this registered as a temp view, it will only be available to this particular notebook. If you'd like other
users to be able to query this table, you can also create a table from the DataFrame.
# Once saved, this table will persist across cluster restarts as well as allow various users across different
notebooks to query this data.
# To do so, choose your table name and uncomment the bottom line.

permanent_table_name = "Credit_csv"

df.write.format("parquet").saveAsTable(permanent_table_name)
```

(1) Spark Jobs

1

17 hours ago (1s) 6 SQL

```
%sql
select * from default.credit_csv
```

(1) Spark Jobs

_sqldf: pyspark.sql.dataframe.DataFrame = [Credit_id: integer, Credit_Name: string ... 2 more fields]

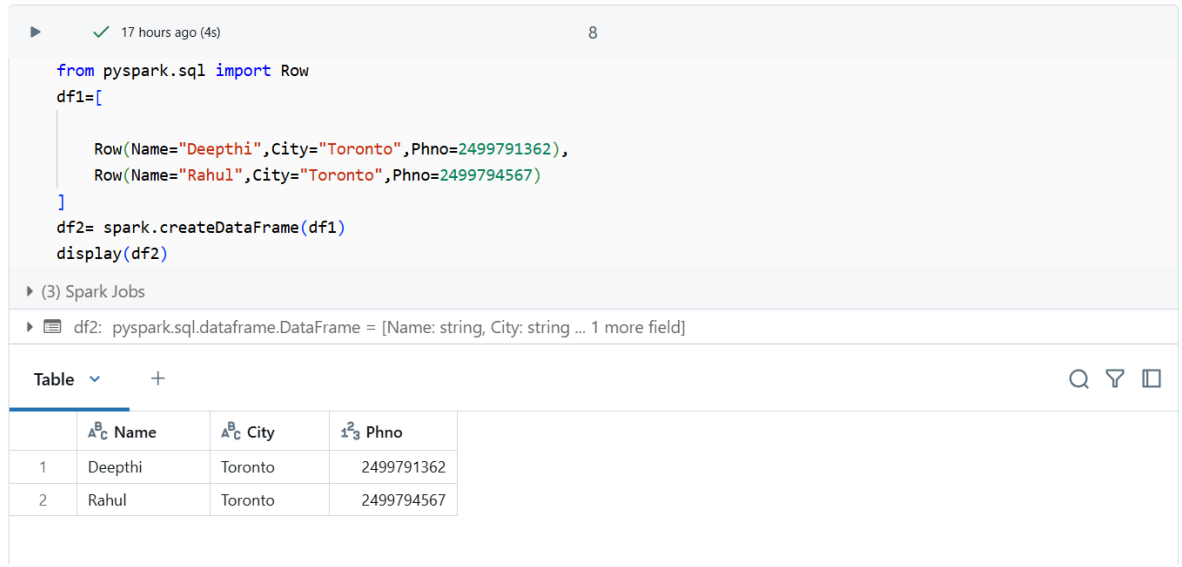
Table

	Credit_id	Credit_Name	Credit_...	Credit_Score
1	1	Deepthi	Master	989
2	2	Rahul	Visa	1124
3	3	Anjana	Visa	1124
4	4	Shriya	Master	765
5	1	Deepthi	Visa	678
6	2	Rahul	Master	1245
7	3	Anjana	Master	989
8	4	Shriya	Visa	765
9	1	Deepthi	Master	989
10	3	Anjana	Master	789

10 rows | 1.42s runtime

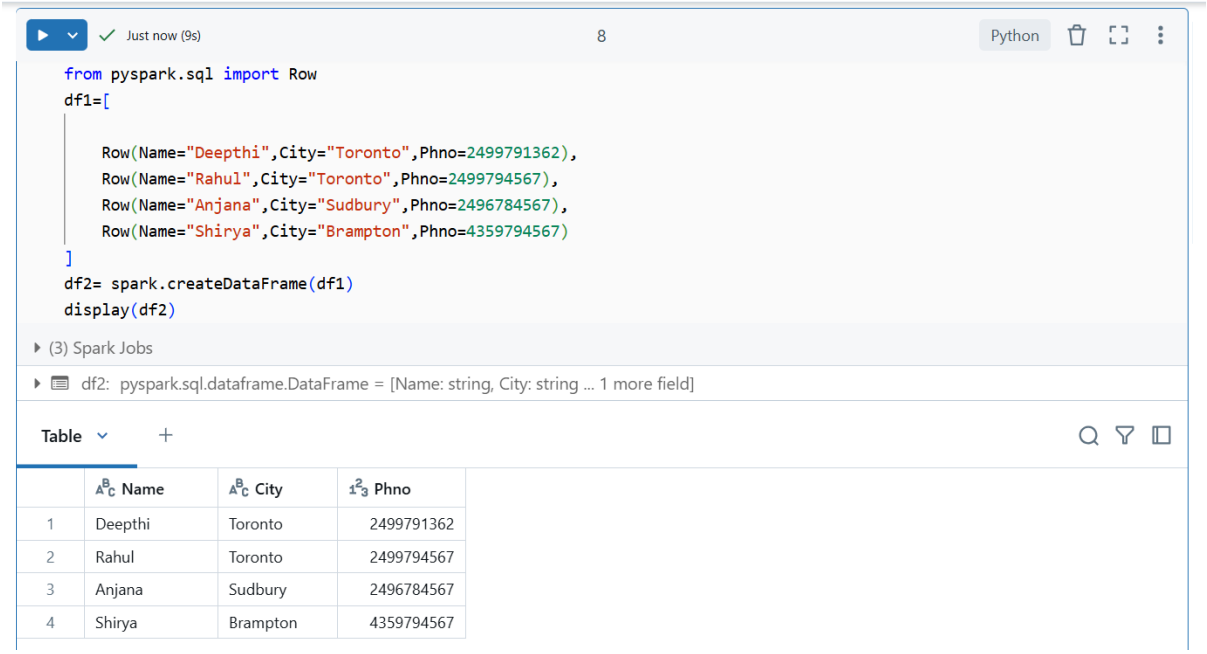
Refreshed 17 hours ago

Create your own data frames using sample data

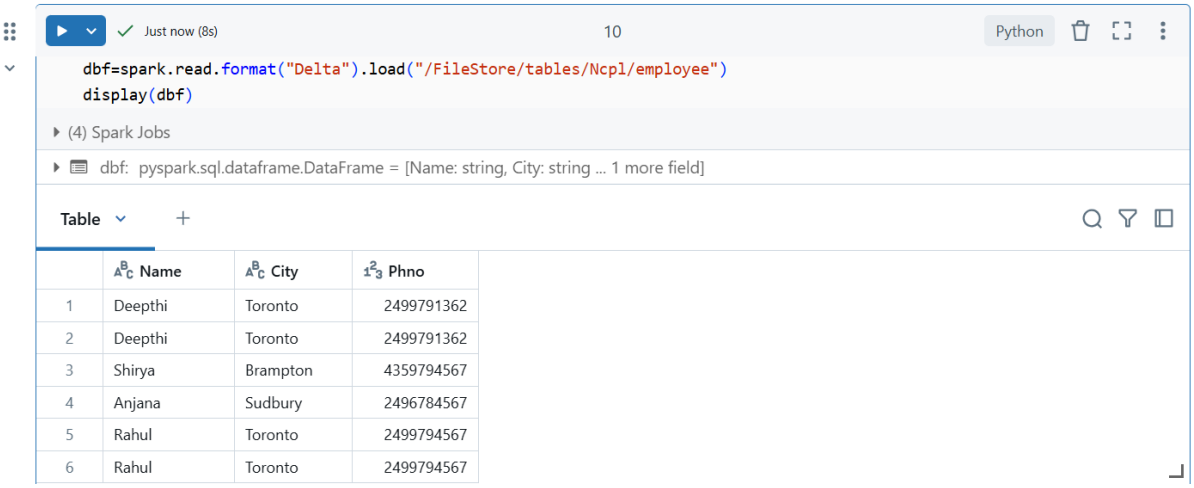


Explore the delta table and version control mechanism

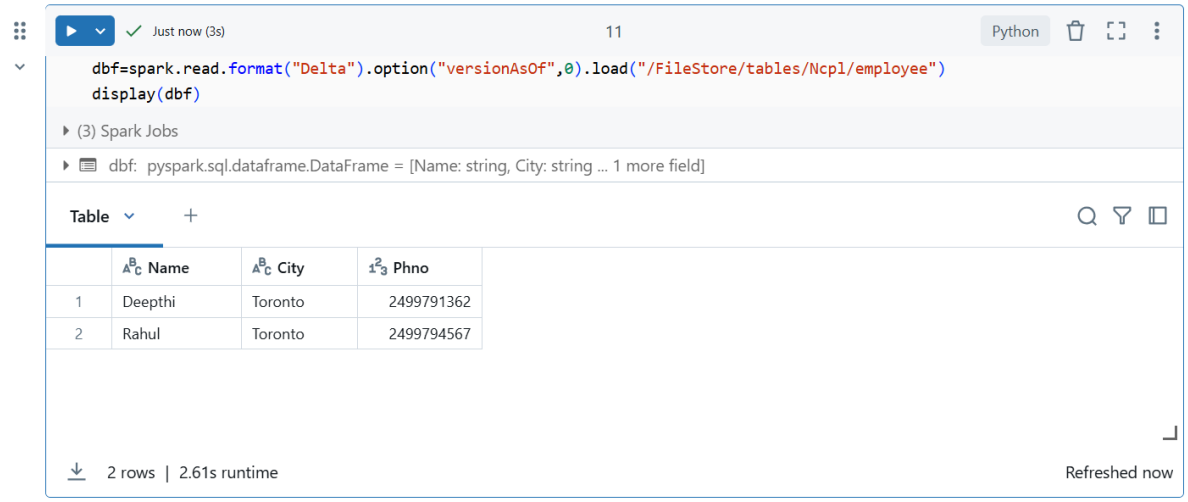
Added new data to data frame

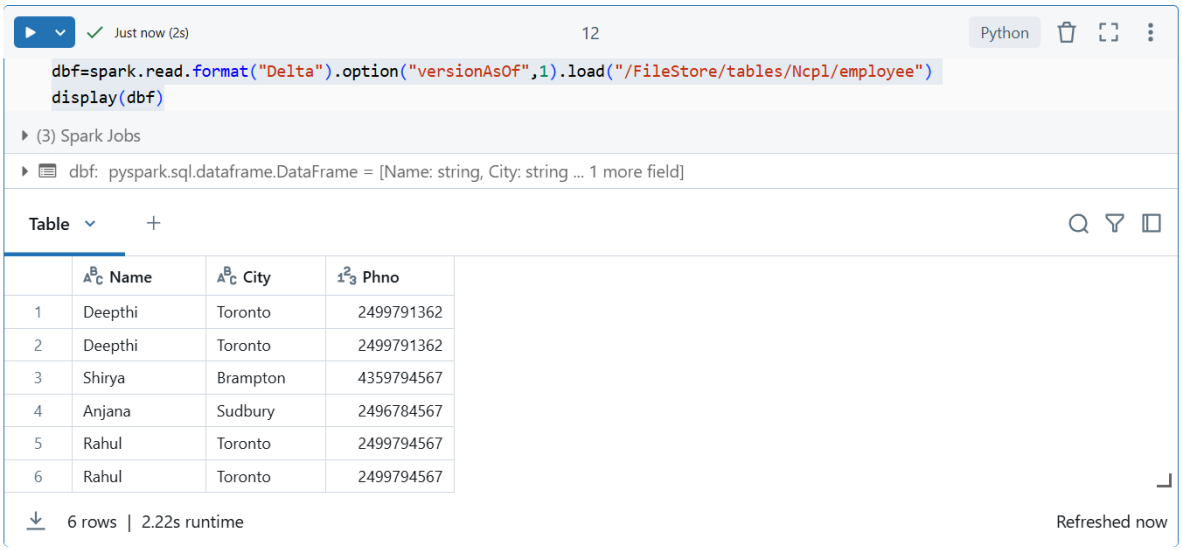


Data is appended

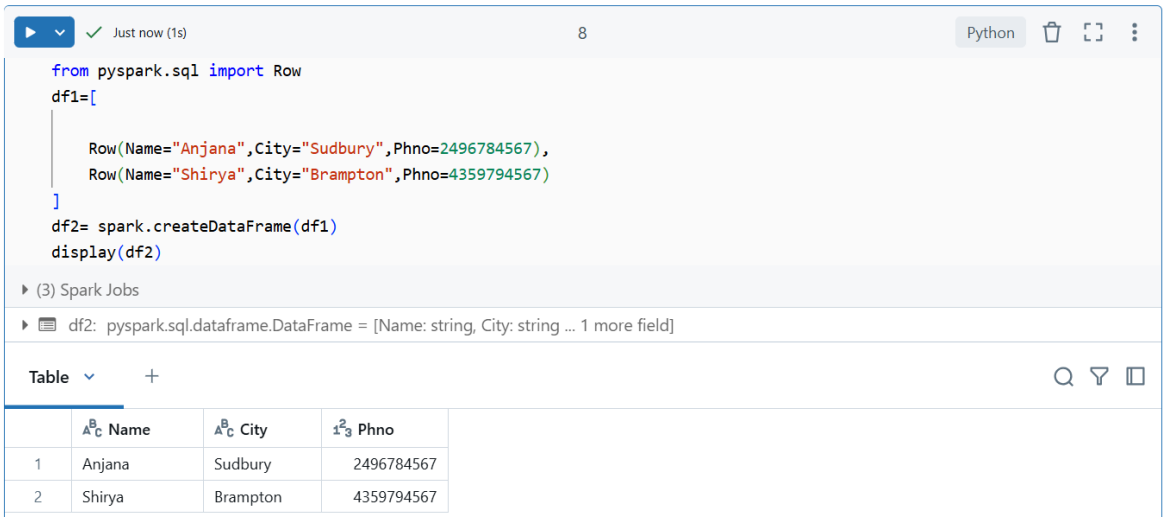


Even after the data changed in data frame if we want first version of data we can get it by using versionAsof





Adding new data and overwriting it



Even though we overwrite the data we can get previous version of data