CRUD Operations

Create Table

Load data into Delta format so it supports ACID transactions and time travel. You can create from CSV/Parquet using PySpark or SQL.

```
df_raw =
spark.read.csv('dbfs:/Volumes/databricks_catalog/default/source/Sales.csv',
header=True, inferSchema=True)
delta_path = 'dbfs:/Volumes/databricks_catalog/default/destination/sales'
df_raw.write.format('delta')\
            .mode('overwrite')\
            .save(delta_path)
```

```
CREATE DATABASE deltaDB;
```

```
CREATE TABLE default.delta_sales
USING DELTA
```

Upsert into the Delta Table

```
updates_data = [
    ("FDA15", 9.3, "Low Fat", 0.016047301, "Dairy", 255.0, "OUT049", 1999,
"Medium", "Tier 1", "Supermarket Type1", 4000.0),  # updated sales
    ("NEW01", 12.5, "Regular", 0.02, "Snacks", 150.0, "OUT010", 2010,
"Small", "Tier 2", "Grocery Store", 1200.0)           # new row
]
updates_df = spark.createDataFrame(
    updates_data,
    df_raw.columns
)
```

Output

```
updates_df: pyspark.sql.connect.dataframe.DataFrame
        Item_Identifier: string
        Item_Weight: double
        Item_Fat_Content: string
        Item_Visibility: double
        Item_Type: string
        Item_MRP: double
        Outlet_Identifier: string
        Outlet_Establishment_Year: long
        Outlet_Size: string
        Outlet_Location_Type: string
        Outlet_Type: string
        Item_Outlet_Sales: double
```

```python
from delta.tables import DeltaTable


delta_table = DeltaTable.forPath(spark,delta_path)
(
    delta_table.alias('tgt')
    .merge(
        updates_df.alias('src'),
        "tgt.Item_Identifier = src.Item_Identifier"
    )
    .whenMatchedUpdateAll()
    .whenNotMatchedInsertAll()
    .execute()
)
```

Output

```
DataFrame[num_affected_rows: bigint, num_updated_rows: bigint, num_deleted_rows: bigint, num_inserted_rows: bigint]
```

Read the data

```python
spark.read.format("delta").load(delta_path).display()
```

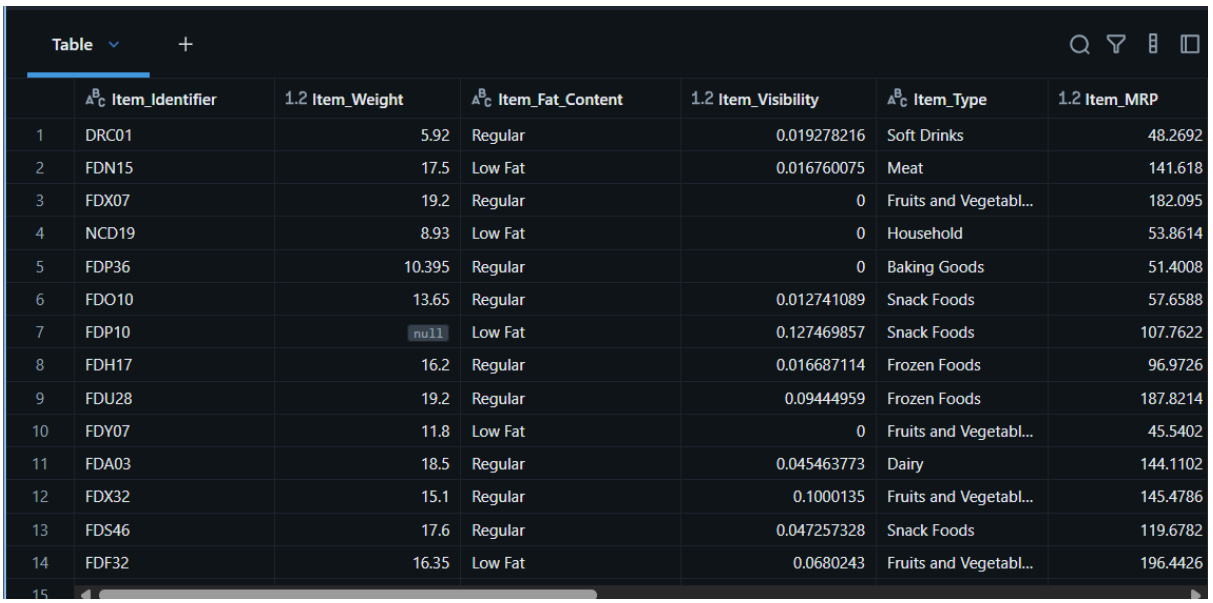

Table +

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP
1	DRC01	5.92	Regular	0.019278216	Soft Drinks	48.2692
2	FDN15	17.5	Low Fat	0.016760075	Meat	141.618
3	FDX07	19.2	Regular	0	Fruits and Vegetabl...	182.095
4	NCD19	8.93	Low Fat	0	Household	53.8614
5	FDP36	10.395	Regular	0	Baking Goods	51.4008
6	FDO10	13.65	Regular	0.012741089	Snack Foods	57.6588
7	FDP10	null	Low Fat	0.127469857	Snack Foods	107.7622
8	FDH17	16.2	Regular	0.016687114	Frozen Foods	96.9726
9	FDU28	19.2	Regular	0.09444959	Frozen Foods	187.8214
10	FDY07	11.8	Low Fat	0	Fruits and Vegetabl...	45.5402
11	FDA03	18.5	Regular	0.045463773	Dairy	144.1102
12	FDX32	15.1	Regular	0.1000135	Fruits and Vegetabl...	145.4786
13	FDS46	17.6	Regular	0.047257328	Snack Foods	119.6782
14	FDF32	16.35	Low Fat	0.0680243	Fruits and Vegetabl...	196.4426
15						

Write to a table

```
df_raw.write.mode("append").saveAsTable("default.delta_sales")
```

Read a table

```
sales_df = spark.read.table("delta_sales")
display(sales_df)
```

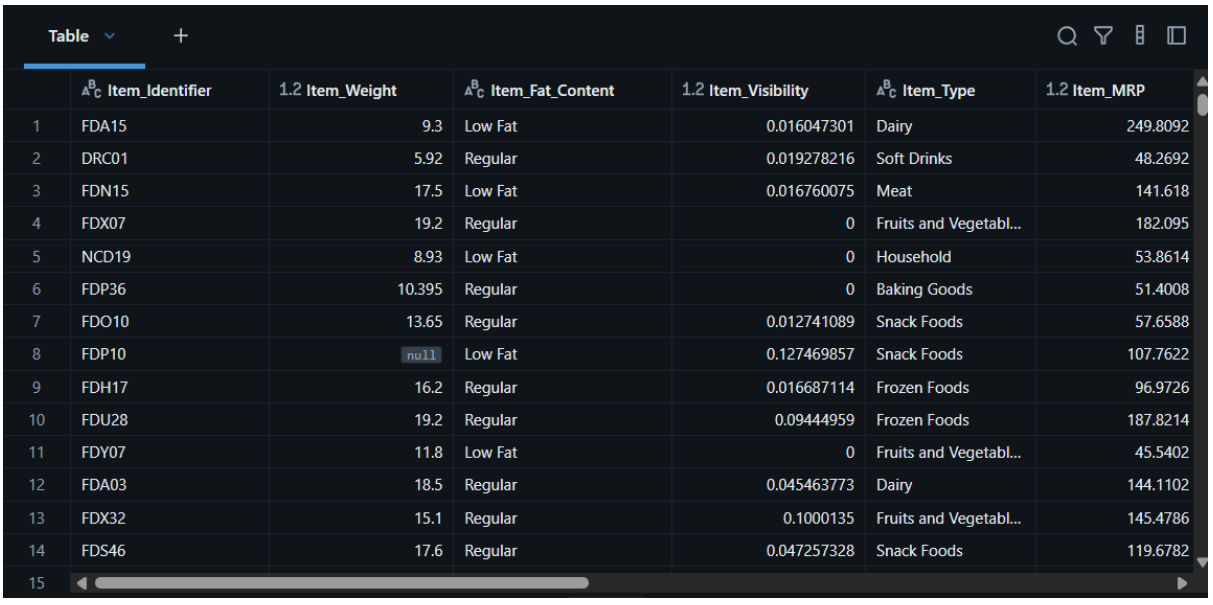

Table +

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP
1	FDA15	9.3	Low Fat	0.016047301	Dairy	249.8092
2	DRC01	5.92	Regular	0.019278216	Soft Drinks	48.2692
3	FDN15	17.5	Low Fat	0.016760075	Meat	141.618
4	FDX07	19.2	Regular	0	Fruits and Vegetabl...	182.095
5	NCD19	8.93	Low Fat	0	Household	53.8614
6	FDP36	10.395	Regular	0	Baking Goods	51.4008
7	FDO10	13.65	Regular	0.012741089	Snack Foods	57.6588
8	FDP10	null	Low Fat	0.127469857	Snack Foods	107.7622
9	FDH17	16.2	Regular	0.016687114	Frozen Foods	96.9726
10	FDU28	19.2	Regular	0.09444959	Frozen Foods	187.8214
11	FDY07	11.8	Low Fat	0	Fruits and Vegetabl...	45.5402
12	FDA03	18.5	Regular	0.045463773	Dairy	144.1102
13	FDX32	15.1	Regular	0.1000135	Fruits and Vegetabl...	145.4786
14	FDS46	17.6	Regular	0.047257328	Snack Foods	119.6782
15						

Update a table

```
from delta.tables import *
from pyspark.sql.functions import *

deltaTable = DeltaTable.forName(spark, "default.delta_sales")
```

```
deltaTable.update(
    set= {
        "Outlet_Location_Type":when(col("Outlet_Location_Type")== "Tier 1",
lit("T1"))
                                .when(col("Outlet_Location_Type")== "Tier 2",
lit("T2"))
                                .when(col("Outlet_Location_Type") == "Tier 3",
lit("T3"))
                                .otherwise(col("Outlet_Location_Type"))
    }
)
```

```
sales_df = spark.read.table("delta_sales")
display(sales_df)
```

Output

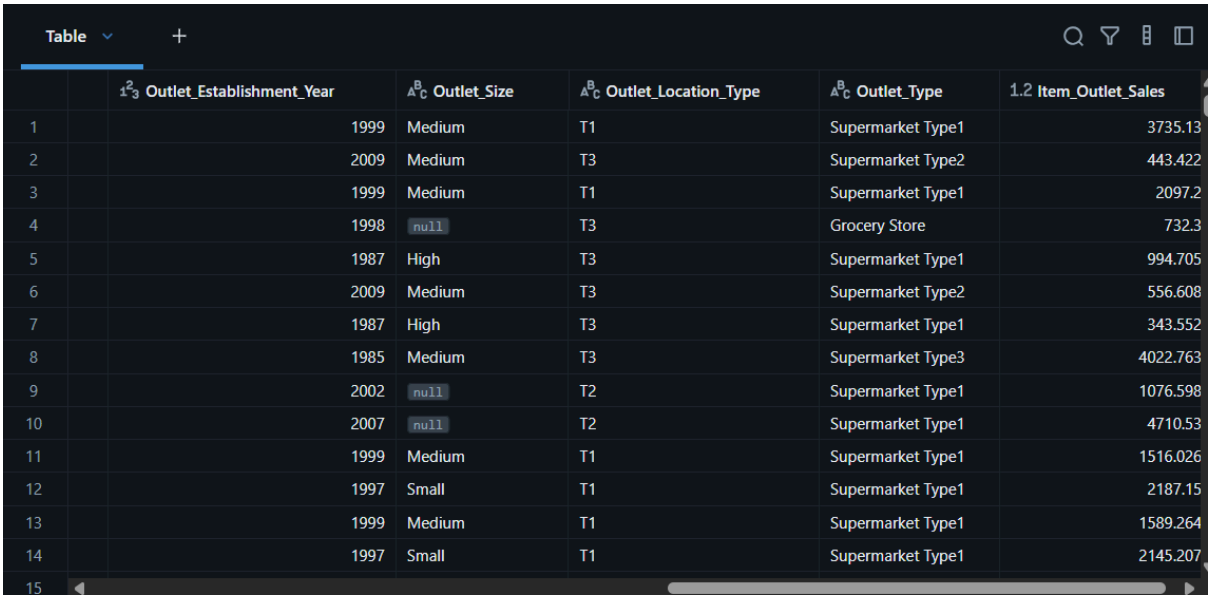

Table +

	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Outlet_Type	Item_Outlet_Sales
1	1999	Medium	T1	Supermarket Type1	3735.13
2	2009	Medium	T3	Supermarket Type2	443.422
3	1999	Medium	T1	Supermarket Type1	2097.2
4	1998	null	T3	Grocery Store	732.3
5	1987	High	T3	Supermarket Type1	994.705
6	2009	Medium	T3	Supermarket Type2	556.608
7	1987	High	T3	Supermarket Type1	343.552
8	1985	Medium	T3	Supermarket Type3	4022.763
9	2002	null	T2	Supermarket Type1	1076.598
10	2007	null	T2	Supermarket Type1	4710.53
11	1999	Medium	T1	Supermarket Type1	1516.026
12	1997	Small	T1	Supermarket Type1	2187.15
13	1999	Medium	T1	Supermarket Type1	1589.264
14	1997	Small	T1	Supermarket Type1	2145.207
15					

Delete from a Table

```
from delta.tables import *
from pyspark.sql.functions import *

deltaTable = DeltaTable.forName(spark, "default.delta_sales")
deltaTable.delete(col("Item_Visibility")=="0")
```

```
sales_df = spark.read.table("delta_sales")
display(sales_df)
```

Output

Table +

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP
1	FDA15	9.3	Low Fat	0.016047301	Dairy	249.8092
2	DRC01	5.92	Regular	0.019278216	Soft Drinks	48.2692
3	FDN15	17.5	Low Fat	0.016760075	Meat	141.618
4	FDO10	13.65	Regular	0.012741089	Snack Foods	57.6588
5	FDP10	null	Low Fat	0.127469857	Snack Foods	107.7622
6	FDH17	16.2	Regular	0.016687114	Frozen Foods	96.9726
7	FDU28	19.2	Regular	0.09444959	Frozen Foods	187.8214
8	FDA03	18.5	Regular	0.045463773	Dairy	144.1102
9	FDX32	15.1	Regular	0.1000135	Fruits and Vegetabl...	145.4786
10	FDS46	17.6	Regular	0.047257328	Snack Foods	119.6782
11	FDF32	16.35	Low Fat	0.0680243	Fruits and Vegetabl...	196.4426
12	FDP49	9	Regular	0.069088961	Breakfast	56.3614
13	NCB42	11.8	Low Fat	0.008596051	Health and Hygiene	115.3492
14	FDP49	9	Regular	0.069196376	Breakfast	54.3614

Display table history

```
from delta.tables import *
deltaTable = DeltaTable.forName(spark, "default.delta_sales")
display(deltaTable.history())
```

Table +

	version	timestamp	userId	userName	operation	operationParame
1	2	2025-08-14T00:51:08.000+00:...	726071949014...	tejashreeganesan04@gmail.co...	DELETE	{"predicate":"[\"(Ite
2	1	2025-08-14T00:47:13.000+00:...	726071949014...	tejashreeganesan04@gmail.co...	UPDATE	{"predicate":"[]"}
3	0	2025-08-14T00:34:27.000+00:...	726071949014...	tejashreeganesan04@gmail.co...	CREATE TABLE AS SELECT	{"partitionBy":"[]","c

Query an earlier version of the table (time travel)

```
describe history delta_sales --get full history of the table
```

Table +

	version	timestamp	userId	userName	operation	operationParamete
1	2	2025-08-14T00:51:08.000+00:...	726071949014...	tejashreeganesan04@gmail.co...	DELETE	{"predicate":"[\"(Iten
2	1	2025-08-14T00:47:13.000+00:...	726071949014...	tejashreeganesan04@gmail.co...	UPDATE	{"predicate":"[]"}
3	0	2025-08-14T00:34:27.000+00:...	726071949014...	tejashreeganesan04@gmail.co...	CREATE TABLE AS SELECT	{"partitionBy":"[]","cl

```
select * from delta_sales timestamp as of '2025-08-14T00:34:27.000+00:00';
```

Output

Table +

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP
1	FDA15	9.3	Low Fat	0.016047301	Dairy	249.8092
2	DRC01	5.92	Regular	0.019278216	Soft Drinks	48.2692
3	FDN15	17.5	Low Fat	0.016760075	Meat	141.618
4	FDX07	19.2	Regular	0	Fruits and Vegetabl...	182.095
5	NCD19	8.93	Low Fat	0	Household	53.8614
6	FDP36	10.395	Regular	0	Baking Goods	51.4008
7	FDO10	13.65	Regular	0.012741089	Snack Foods	57.6588
8	FDP10	null	Low Fat	0.127469857	Snack Foods	107.7622
9	FDH17	16.2	Regular	0.016687114	Frozen Foods	96.9726
10	FDU28	19.2	Regular	0.09444959	Frozen Foods	187.8214
11	FDY07	11.8	Low Fat	0	Fruits and Vegetabl...	45.5402
12	FDA03	18.5	Regular	0.045463773	Dairy	144.1102
13	FDX32	15.1	Regular	0.1000135	Fruits and Vegetabl...	145.4786
14	FDS46	17.6	Regular	0.047257328	Snack Foods	119.6782
15						

```
select * from delta_sales@v2;
```

Table +

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP
6	FDH17	16.2	Regular	0.016687114	Frozen Foods	96.9726
7	FDU28	19.2	Regular	0.09444959	Frozen Foods	187.8214
8	FDA03	18.5	Regular	0.045463773	Dairy	144.1102
9	FDX32	15.1	Regular	0.1000135	Fruits and Vegetabl...	145.4786
10	FDS46	17.6	Regular	0.047257328	Snack Foods	119.6782
11	FDF32	16.35	Low Fat	0.0680243	Fruits and Vegetabl...	196.4426
12	FDP49	9	Regular	0.069088961	Breakfast	56.3614
13	NCB42	11.8	Low Fat	0.008596051	Health and Hygiene	115.3492
14	FDP49	9	Regular	0.069196376	Breakfast	54.3614
15	DRI11	null	Low Fat	0.034237682	Hard Drinks	113.2834
16	FDU02	13.35	Low Fat	0.10249212	Dairy	230.5352
17	FDN22	18.85	Regular	0.138190277	Snack Foods	250.8724
18	FDW12	null	Regular	0.035399923	Baking Goods	144.5444
19	NCB30	14.6	Low Fat	0.025698134	Household	196.5084

```
df1 = spark.read.option("timestampAsOf",
"2025-08-14T00:51:08.000+00:00").table("delta_sales").display()
```

Table +

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP
1	FDA15	9.3	Low Fat	0.016047301	Dairy	249.8092
2	DRC01	5.92	Regular	0.019278216	Soft Drinks	48.2692
3	FDN15	17.5	Low Fat	0.016760075	Meat	141.618
4	FDO10	13.65	Regular	0.012741089	Snack Foods	57.6588
5	FDP10	null	Low Fat	0.127469857	Snack Foods	107.7622
6	FDH17	16.2	Regular	0.016687114	Frozen Foods	96.9726
7	FDU28	19.2	Regular	0.09444959	Frozen Foods	187.8214
8	FDA03	18.5	Regular	0.045463773	Dairy	144.1102
9	FDX32	15.1	Regular	0.1000135	Fruits and Vegetabl...	145.4786
10	FDS46	17.6	Regular	0.047257328	Snack Foods	119.6782
11	FDF32	16.35	Low Fat	0.0680243	Fruits and Vegetabl...	196.4426
12	FDP49	9	Regular	0.069088961	Breakfast	56.3614
13	NCB42	11.8	Low Fat	0.008596051	Health and Hygiene	115.3492
14	FDP49	9	Regular	0.069196376	Breakfast	54.3614
15						

Restore a Delta table to an earlier state

```
delete from delta_sales
where Item_Identifier = 'FDH17';
```

Table +

	num_affected_rows
1	7