≫ Delta Lake tables provide ACID compliant updates to tables.

≫ OVERWRITING TABLES :-

≫ Much faster than deleting and recreating the table.

≫ Old version still exists, and can be retrieved by Time Travel.

① Using CTAS to completely overwrite a tables' data records :-

```
CREATE OR REPLACE TABLE <table_name> AS
SELECT * FROM parquet.`<location>`
```

Above CRAS statement can do both :-

a. Create a new table (if non-existent)
b. Overwrite data records of table (if existent)

② INSERT OVERWRITE to overwrite data :-

```
INSERT OVERWRITE <table_to_be_overwritten>
SELECT * FROM parquet.`<location>`
```

any source / type of data : parquet, json

Properties :-

a) Can only overwrite data if table exists : cannot create new tables.

b) Will overwrite only those new records, that match the current table schema, and thus is a safer way of overwriting (if we don't want to disrupt existing table.)

c) Can overwrite individual partitions.

Table history for this command is recorded differently.

2 Appending New/Updated Rows :-

Syntax :-

table in which we want to append rows

```
INSERT INTO <table name>
SELECT * FROM parquet.`<location>`
```

data files containing new records, that we want to append.

a) Automatically appends new rows to existing Delta table.

b) Allows incremental updates. More efficient than overwriting repeatedly.

- Can result in duplicate record entry, if run multiple times.

Upserting Data using MERGE

- Completes insert, updates & deletes in a single transaction.
- Multiple conditions can be added in addition to matching fields.

Ex:-

```
MERGE INTO users a
USING users_update b
ON a.user_id = b.user_id
WHEN MATCHED AND a.email IS NULL AND b.email IS NOT NULL THEN
    UPDATE SET email = b.email, updated = b.updated
WHEN NOT MATCHED THEN
    INSERT *
```

Updating records if current row has NULL and new row does not.

All unmatched records will be inserted.

Appending only Non-Duplicate rec:-

- Can be done using MERGE command, and only passing the NOT MATCHED condition:-

Insert-Only Merge for Deduplication

A common ETL use case is to collect logs or other every-appending datasets into a Delta table through a series of append operations. Many source systems can generate duplicate records. With merge, you can avoid inserting the duplicate records by performing an insert-only merge.

This optimized command uses the same `MERGE` syntax but only provided a `WHEN NOT MATCHED` clause.

Below, we use this to confirm that records with the same `user_id` and `event_timestamp` aren't already in the `events` table.

nd 25

```
1  MERGE INTO events a
2  USING events_update b
3  ON a.user_id = b.user_id AND a.event_timestamp = b.event_timestamp
4  WHEN NOT MATCHED AND b.traffic_source = 'email' THEN
5    INSERT *
```

Loading Incrementally :-

table in which we want to append.

```
COPY INTO <table_name>
FROM "dbfs:/FileStore/folder/filename"
FILEFORMAT = PARQUET
```

in real life this location for the command will change everytime - will be dynamic

- Loads incrementally.
- Data schema should be consistent.
- Duplicate records should be dried to be excluded / removed manually.
- This operation is much cheaper than full table scans.