Delta tables

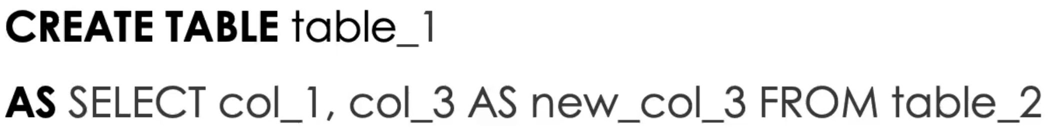
- CTAS statements (**CREATE TABLE AS SELECT**) are used to create delta tables.

  CREATE TABLE table_1

  AS **SELECT** * FROM table_2

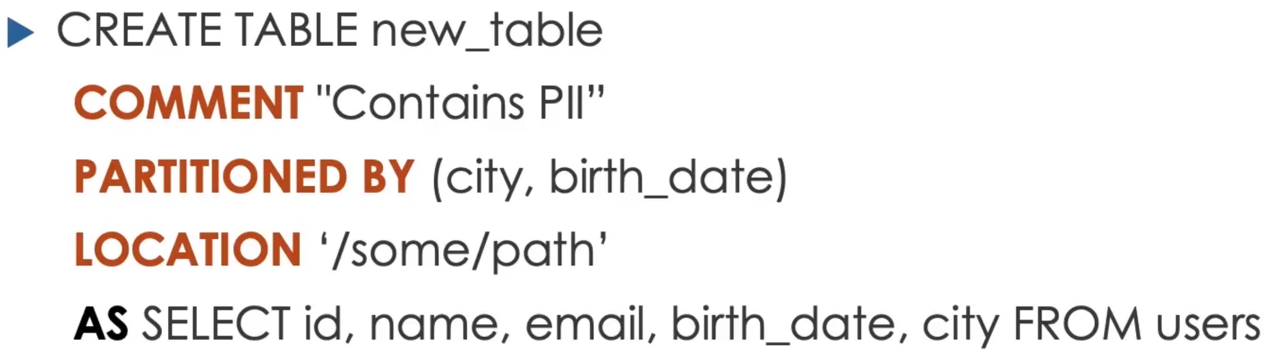
  table_1 will be created as data retrieved from table_2.

- Automatically infer schema information from query results, it don't support manual schema declaration.
- We can filter and rename columns from input table using CTAS statements.



```
CREATE TABLE table_1
AS SELECT col_1, col_3 AS new_col_3 FROM table_2
```

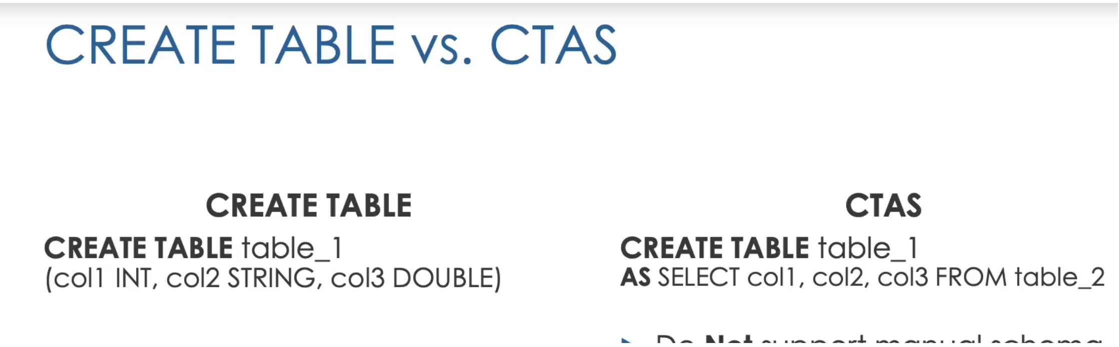
- Additional options like comment, partitioned by one or more columns, location can be used in CTAS statement.



```
▶ CREATE TABLE new_table
   COMMENT "Contains PII"
   PARTITIONED BY (city, birth_date)
   LOCATION '/some/path'
   AS SELECT id, name, email, birth_date, city FROM users
```

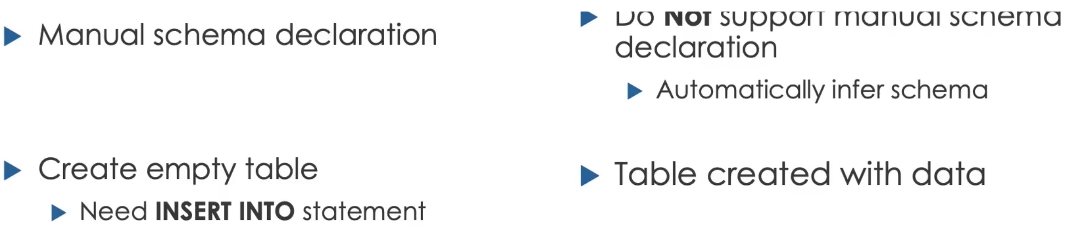
- Partitioning can improve performance of huge delta tables. It doesn't benefit small-medium sized tables because partitioning physically separates data files which results in small files a problem which can prevent file compaction and efficient data skipping.

- Default to non-partition tables when working in most of the cases when working with Delta tables.



CREATE TABLE vs. CTAS

CREATE TABLE	CTAS
CREATE TABLE table_1 (col1 INT, col2 STRING, col3 DOUBLE)	**AS** SELECT col1, col2, col3 FROM table_2
	▸ Do **Not** support manual schema



- Manual schema declaration
- Create empty table
  - Need **INSERT INTO** statement

- Do **Not** support manual schema declaration
  - Automatically infer schema
- Table created with data

- Databricks supports two types of table constraints - **NOT NULL & CHECK**.

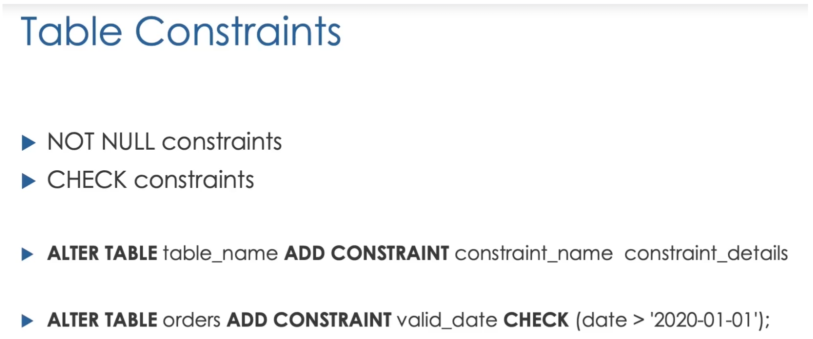
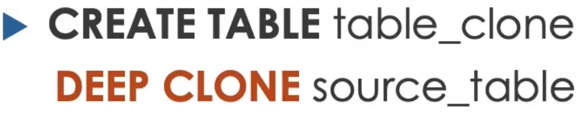


Table Constraints

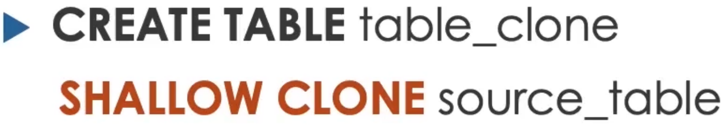
- NOT NULL constraints
- CHECK constraints
- **ALTER TABLE** table_name **ADD CONSTRAINT** constraint_name constraint_details
- **ALTER TABLE** orders **ADD CONSTRAINT** valid_date **CHECK** (date > '2020-01-01');

- Cloning Delta lake tables - **Deep Clone & Shallow Clone**
- Deep cloning fully copies data and metadata from source table into target table, can sync changes. Take quite long when cloning large datasets.



```
CREATE TABLE table_clone
  DEEP CLONE source_table
```

- Shallow cloning quickly create copy of table, just copy delta transaction logs. No data moving during shallow cloning. Good option to test out applying changes on table without risk of modifying current table.



```
CREATE TABLE table_clone
  SHALLOW CLONE source_table
```

- Cloning is a great way to setup tables for testing in development. In either cases, data modifications don't affect the source table.