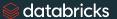
Schema enforcement and evolution



# Lesson goals

- Explain the benefits of schema enforcement and evolution.
- Demonstrate the use of schema enforcement and evolution with Delta tables.



#### Schema enforcement and evolution

- Schema enforcement (aka validation): prevents bad writes
- Schema evolution: allows for updates to data schema
- Delta saves table schema in JSON format in the transaction log
- Schema evolution commonly used for appends or overwrites



#### Schema enforcement

- Enforced on tables that directly feed:
  - Machine learning algorithms
  - Bl dashboards
  - Data analytics and visualization tools
  - Production system requiring highly structured schemas



#### Schema enforcement

- Dataframe to write cannot:
  - Contain additional columns not present in the target table's schema
  - Have column data types that differ from the column data types in the target table
  - Contain column names that differ only by case



### Schema enforcement in action

```
( new_health_tracker_data_df.write
         .format("delta")
          .mode("append")
         .save(schema_dir) )
  HanalysisException: A schema mismatch detected when writing to the Delta table (Table ID: 94ea1294-9d88-43
   76-bc35-0727970a4280).
HanalysisException: A schema mismatch detected when writing to the Delta table (Table ID: 94ea1294-9d88-43
                                Table schema:
  76-bc35-0727970a4280).
                                root
                                -- device id: long (nullable = true)
                                -- heartrate: double (nullable = true)
                                -- name: string (nullable = true)
                                -- time: double (nullable = true)
                                Data schema:
                                root
                                -- brand: string (nullable = true)
                                -- device_id: long (nullable = true)
                                -- heartrate: double (nullable = true)
                                -- name: string (nullable = true)
                                -- time: double (nullable = true)
databricks
```

### Schema evolution

#### Eligible for schema evolution



- Adding new columns
- Changing data types from:
  - NullType → any other type
  - ByteType → ShortType → IntegerType
- Use .option ("mergeSchema", "true")

#### Not eligible for schema evolution



- Dropping a column
- Changing an existing column's data type
- Renaming column names that differ only by case
- Must use .option ("overwriteSchema", "true")



## Schema evolution in action

```
( new_health_tracker_data_df.write
       .format("delta")
       .option("mergeSchema", "true")
       .mode("append")
       .save(schema_dir) )
HanalysisException: Failed to merge fields 'device_id' and 'device_id'. Failed to merge incompatible data types
 LongType and StringType;;
                                                                                                                 94ea1294-9d88-43
76-bc35-0727970a4280).
   ( new_health_tracker_data_df.write
        format("delta")
       .option("overwriteSchema", "true")
       .mode("append")
       .save(schema_dir) )
```



# **databricks**