```
# etl/load_bts.py
import os, pandas as pd
from sqlalchemy import create_engine, text
from dotenv import load_dotenv

load_dotenv()
engine = create_engine(os.environ["AIRLINE_DB_DSN"], pool_pre_ping=True)

def clean_str(s):
    if pd.isna(s): return None
    s = str(s).strip()
    return s or None

def run():
    # Start with a small subset CSV (1–3 months)
    bts = pd.read_csv("data/bts_flights_2024Q1.csv")

    # Map common BTS columns -> normalized names
    rename_map = {
        "MKT_UNIQUE_CARRIER":"airline_code",
        "OP_UNIQUE_CARRIER":"airline_code",   # fallback
        "MKT_CARRIER_FL_NUM":"flight_number",
        "FL_DATE":"flight_date",
        "ORIGIN":"origin_iata",
        "DEST":"dest_iata",
        "CRS_DEP_TIME_UTC":"sched_dep_utc",
        "CRS_ARR_TIME_UTC":"sched_arr_utc",
        "DEP_TIME_UTC":"actual_dep_utc",
        "ARR_TIME_UTC":"actual_arr_utc",
        "ARR_DELAY":"delay_minutes",
        "CANCELLED":"cancelled"
    }
    bts = bts.rename(columns={c:rename_map.get(c,c) for c in bts.columns})

    # Clean/convert types
    for c in ["airline_code","flight_number","origin_iata","dest_iata"]:
        if c in bts.columns: bts[c] = bts[c].map(clean_str)
    bts["flight_date"] = pd.to_datetime(bts["flight_date"], errors="coerce").dt.date
    for col in ["sched_dep_utc","sched_arr_utc","actual_dep_utc","actual_arr_utc"]:
        if col in bts.columns: bts[col] = pd.to_datetime(bts[col], errors="coerce")
    bts["delay_minutes"] = pd.to_numeric(bts.get("delay_minutes"), errors="coerce")
    # Normalize status
    if "cancelled" in bts.columns:
        bts["status"] = bts["cancelled"].map({1:"Cancelled", 0:"Arrived"})
    bts["status"] = bts["status"].fillna("Scheduled")

    with engine.begin() as con:
        con.execute(text("""
            CREATE TEMP TABLE tmp_bts(
              airline_code text,
              flight_number text,
              flight_date date,
              origin_iata text,
              dest_iata text,
              sched_dep_utc timestamp,
              sched_arr_utc timestamp,
              actual_dep_utc timestamp,
              actual_arr_utc timestamp,
              delay_minutes int,
              status text
            ) ON COMMIT DROP;
        """))
```

```
        cols = ["airline_code","flight_number","flight_date","origin_iata","dest_iata",
                "sched_dep_utc","sched_arr_utc","actual_dep_utc","actual_arr_utc",
                "delay_minutes","status"]
        # keep only existing columns
        cols = [c for c in cols if c in bts.columns]
        bts[cols].to_sql("tmp_bts", con, index=False, if_exists="append")

        con.execute(text("""
            INSERT INTO airline.flights(
              airline_id, aircraft_id, route_id,
              origin_airport_id, destination_airport_id,
              flight_number, flight_date,
              scheduled_departure_utc, scheduled_arrival_utc,
              actual_departure_utc, actual_arrival_utc,
              delay_minutes, delay_cause, status
            )
            SELECT
              al.airline_id,
              NULL::bigint AS aircraft_id,
              r.route_id,
              ao.airport_id, ad.airport_id,
              t.flight_number, t.flight_date,
              t.sched_dep_utc, t.sched_arr_utc,
              t.actual_dep_utc, t.actual_arr_utc,
              t.delay_minutes,
              NULL::text AS delay_cause,
              CASE
                WHEN LOWER(t.status) IN
('scheduled','departed','arrived','cancelled','diverted')
                  THEN INITCAP(LOWER(t.status))
                ELSE 'Scheduled'
              END::flight_status
            FROM tmp_bts t
            JOIN airline.airlines al ON al.iata_code = TRIM(t.airline_code)
            JOIN airline.airports ao ON ao.iata_code = TRIM(t.origin_iata)
            JOIN airline.airports ad ON ad.iata_code = TRIM(t.dest_iata)
            LEFT JOIN airline.routes r
              ON r.airline_id = al.airline_id
             AND r.origin_airport_id = ao.airport_id
             AND r.destination_airport_id = ad.airport_id
            ON CONFLICT ON CONSTRAINT uq_flight_instance DO NOTHING;
        """))

    print("BTS -> production flights complete.")

if __name__ == "__main__":
    run()
```