**Queries:**

**Raw csv to stage:**

CREATE DATABASE data\_design;

USE DATABASE data\_design;

CREATE SCHEMA staging;

USE SCHEMA staging;

-- 1. Create a file format for CSV

CREATE OR REPLACE FILE FORMAT csv\_format

TYPE = 'CSV'

FIELD\_OPTIONALLY\_ENCLOSED\_BY = '"'

FIELD\_DELIMITER = ','

SKIP\_HEADER = 1

NULL\_IF = ('NULL', 'null');

-- 2. Create a stage using CSV format

CREATE OR REPLACE STAGE csv\_stage

FILE\_FORMAT = csv\_format;

list @csv\_stage;

-- 3. Create table for precipitation data

CREATE OR REPLACE TABLE lv\_precipitation (

date DATE,

precipitation STRING,

precipitation\_normal STRING

);

-- 4. Create table for temperature data

CREATE OR REPLACE TABLE lv\_temperature (

date DATE,

min DOUBLE,

max DOUBLE,

normal\_min DOUBLE,

normal\_max DOUBLE

);

COPY INTO lv\_precipitation

FROM @csv\_stage/USW00093134-LOS\_ANGELES\_DOWNTOWN\_USC-precipitation-inch.csv.gz

FILE\_FORMAT = (FORMAT\_NAME = csv\_format);

COPY INTO lv\_temperature

FROM @csv\_stage/USW00093134-temperature-degreeF.csv.gz

FILE\_FORMAT = (FORMAT\_NAME = csv\_format);

select \* from lv\_temperature;

select \* from lv\_precipitation;

**Raw Json to Stage:**

USE DATABASE data\_design;

USE SCHEMA staging;

-- Create a file format for JSON files

create or replace file format jsonformat type='JSON' strip\_outer\_array=true;

-- Create a stage for JSON files using the defined format

create or replace stage json\_stage file\_format = jsonformat;

list @json\_stage;

--REMOVE @json\_stage;

create or replace table yelp\_academic\_dataset\_business(recordjson variant);

create or replace table yelp\_academic\_dataset\_checkin(recordjson variant);

create or replace table yelp\_academic\_dataset\_covid\_features(recordjson variant);

create or replace table yelp\_academic\_dataset\_review(recordjson variant);

create or replace table yelp\_academic\_dataset\_tip(recordjson variant);

create or replace table yelp\_academic\_dataset\_user(recordjson variant

);

USE WAREHOUSE my\_wh;

-- Load business data

COPY INTO yelp\_academic\_dataset\_business

FROM @json\_stage/yelp\_academic\_dataset\_business.json.gz

FILE\_FORMAT = (FORMAT\_NAME = jsonformat);

-- Load checkin data

COPY INTO yelp\_academic\_dataset\_checkin

FROM @json\_stage/yelp\_academic\_dataset\_checkin.json.gz

FILE\_FORMAT = (FORMAT\_NAME = jsonformat);

-- Load covid features data

COPY INTO yelp\_academic\_dataset\_covid\_features

FROM @json\_stage/yelp\_academic\_dataset\_covid\_features.json.gz

FILE\_FORMAT = (FORMAT\_NAME = jsonformat);

-- Load review data

COPY INTO yelp\_academic\_dataset\_review

FROM @json\_stage/yelp\_academic\_dataset\_review.json.gz

FILE\_FORMAT = (FORMAT\_NAME = jsonformat);

-- Load tip data

COPY INTO yelp\_academic\_dataset\_tip

FROM @json\_stage/yelp\_academic\_dataset\_tip.json.gz

FILE\_FORMAT = (FORMAT\_NAME = jsonformat);

-- Load user data

COPY INTO yelp\_academic\_dataset\_user

FROM @json\_stage/yelp\_academic\_dataset\_user.json.gz

FILE\_FORMAT = (FORMAT\_NAME = jsonformat);

select \* from yelp\_academic\_dataset\_business

limit 50;

SELECT

recordjson:"business\_id"::STRING AS business\_id,

recordjson:"name"::STRING AS name,

recordjson:"address"::STRING AS address,

recordjson:"city"::STRING AS city,

recordjson:"state"::STRING AS state,

recordjson:"postal\_code"::STRING AS postal\_code,

recordjson:"latitude"::FLOAT AS latitude,

recordjson:"longitude"::FLOAT AS longitude,

recordjson:"stars"::FLOAT AS stars,

recordjson:"review\_count"::INT AS review\_count,

recordjson:"is\_open"::INT AS is\_open,

recordjson:"categories"::STRING AS categories,

recordjson:"attributes":"ByAppointmentOnly"::STRING AS by\_appointment\_only

FROM yelp\_academic\_dataset\_business

limit 1;

**Staging Data to ODS**:

USE DATABASE data\_design;

create schema ODS;

USE SCHEMA ODS;

-- Create a medium warehouse

CREATE OR REPLACE WAREHOUSE my\_wh

WAREHOUSE\_SIZE = 'MEDIUM'

AUTO\_SUSPEND = 60 -- suspend after 60 seconds of inactivity

AUTO\_RESUME = TRUE -- resume automatically when a query runs

MIN\_CLUSTER\_COUNT = 1 -- autoscaling min

MAX\_CLUSTER\_COUNT = 3 -- autoscaling max

SCALING\_POLICY = 'STANDARD';

-- Use the warehouse

USE WAREHOUSE my\_wh;

-- GEOGRAPHY TABLE

create or replace table "GEOGRAPHY"(

"geography\_id" number identity primary key,

"address" string,

"latitude" double,

"longitude" double,

"postal\_code" string,

"city" string,

"state" string

);

insert into "GEOGRAPHY"("address", "latitude", "longitude", "postal\_code", "city", "state")

select distinct

RECORDJSON:address,

RECORDJSON:latitude,

RECORDJSON:longitude,

RECORDJSON:postal\_code,

RECORDJSON:city,

RECORDJSON:state

from

data\_design.staging.YELP\_ACADEMIC\_DATASET\_BUSINESS;

-- BUSINESS TABLE

create or replace table "BUSINESS"(

"business\_id" string primary key,

"geography\_id" number references data\_design.ODS."GEOGRAPHY"("geography\_id"),

"name" string,

"is\_open" string,

"stars" double

);

insert into "BUSINESS"

select distinct

RECORDJSON:business\_id,

g."geography\_id",

RECORDJSON:name,

RECORDJSON:is\_open,

RECORDJSON:stars

from

data\_design.staging.YELP\_ACADEMIC\_DATASET\_BUSINESS as b

join

data\_design.ODS."GEOGRAPHY" as g

on

RECORDJSON:city = g."city" and

RECORDJSON:address = g."address" and

RECORDJSON:latitude = g."latitude" and

RECORDJSON:longitude = g."longitude" and

RECORDJSON:state = g."state";

-- CHECKIN TABLE

create or replace table "CHECKIN"(

"business\_id" string primary key references data\_design.staging."BUSINESS"("business\_id"),

"date" string

);

insert into "CHECKIN"

select

RECORDJSON:business\_id,

RECORDJSON:date

from

data\_design.staging.YELP\_ACADEMIC\_DATASET\_CHECKIN;

-- CUSTOMER TABLE

create or replace table "CUSTOMER"(

"customer\_id" string primary key,

"average\_stars" double,

"fans" number,

"review\_count" number,

"name" string

);

insert into "CUSTOMER"

select

RECORDJSON:user\_id,

RECORDJSON:average\_stars,

RECORDJSON:fans,

RECORDJSON:review\_count,

RECORDJSON:name

from

data\_design.staging.YELP\_ACADEMIC\_DATASET\_USER;

-- COVID TABLE

create or replace table "COVID"(

"business\_id" string primary key references data\_design.ODS."BUSINESS"("business\_id"),

"call\_action" string,

"covid\_banner" string,

"grubhub" string,

"request\_a\_quote" string,

"temporary\_closed" string,

"virtual\_services" string,

"delivery\_or\_takeout" string,

"highlights" string

);

insert into "COVID"

select

RECORDJSON:"Call To Action enabled",

RECORDJSON:"Covid Banner",

RECORDJSON:"Grubhub enabled",

RECORDJSON:"Request a Quote Enabled",

RECORDJSON:"Temporary Closed Until",

RECORDJSON:"Virtual Services Offered",

RECORDJSON:business\_id,

RECORDJSON:"delivery or takeout",

RECORDJSON:highlights

from

data\_design.staging.YELP\_ACADEMIC\_DATASET\_COVID\_FEATURES;

create or replace table "REVIEW"(

"review\_id" string primary key,

"business\_id" string references data\_design.ODS."BUSINESS"("business\_id"),

"date" date,

"cool" number,

"funny" number,

"stars" double,

"useful" double,

"user\_id" string references data\_design.ODS."CUSTOMER"("customer\_id")

);

-- REVIEW TABLE

insert into "REVIEW"

select

RECORDJSON:review\_id,

RECORDJSON:business\_id,

RECORDJSON:date::date,

RECORDJSON:cool,

RECORDJSON:funny,

RECORDJSON:stars,

RECORDJSON:useful,

RECORDJSON:user\_id

from

data\_design.staging.YELP\_ACADEMIC\_DATASET\_REVIEW;

-- TIP TABLE

create or replace table "TIP"(

"business\_id" string primary key references data\_design.ODS."BUSINESS"("business\_id"),

"compliment\_count" number,

"date" date,

"user\_id" string references data\_design.ODS."CUSTOMER"("customer\_id")

);

insert into "TIP"

select

RECORDJSON:business\_id,

RECORDJSON:compliment\_count,

RECORDJSON:date,

RECORDJSON:user\_id

from

data\_design.staging.YELP\_ACADEMIC\_DATASET\_TIP;

-- TEMPERATURE TABLE

create or replace table "TEMPERATURE"(

"date" date primary key,

"min\_temp" double,

"max\_temp" double,

"normal\_min" double,

"normal\_max" double

);

insert into "TEMPERATURE"(

"date", "min\_temp", "max\_temp", "normal\_min", "normal\_max"

)

select

date Date,

min double,

max double,

normal\_min double,

normal\_max double

from

data\_design.staging.lv\_temperature;

-- PRECIPITATION TABLE

create or replace table "PRECIPITATION"(

"date" date primary key,

"precipitation" string,

"precipitation\_normal" double

);

insert into "PRECIPITATION"(

"date", "precipitation", "precipitation\_normal"

)

select

date,

precipitation,

precipitation\_normal

from

data\_design.staging.lv\_precipitation;

**ODS data to DWH:**

Use database data\_design;

create schema DWHH;

USE SCHEMA DWHH;

USE WAREHOUSE my\_wh;

drop schema DWHH;

create or replace table "DIM\_CUSTOMER"(

"customer\_id" string primary key,

"name" string,

"average\_stars" double,

"fans" number,

"review\_count" number

);

insert into "DIM\_CUSTOMER"

select distinct

"customer\_id",

"name",

"average\_stars",

"fans",

"review\_count"

from data\_design.ODS."CUSTOMER";

-- table dim\_temperature

create or replace table "DIM\_TEMPERATURE"(

"date" DATE primary key,

"min\_temp" double,

"max\_temp" double,

"normal\_min" double,

"normal\_max" double

);

insert into "DIM\_TEMPERATURE"

select distinct

"date",

"min\_temp",

"max\_temp",

"normal\_min",

"normal\_max"

from

data\_design.ODS."TEMPERATURE";

-- table dim\_precipitation

create or replace table "DIM\_PRECIPITATION"(

"date" DATE primary key,

"precipitation" string,

"precipitation\_normal" string

);

insert into "DIM\_PRECIPITATION"

select distinct

"date",

"precipitation",

"precipitation\_normal"

from

data\_design.ODS."PRECIPITATION";

-- table dim\_business

create or replace table "DIM\_BUSINESS"(

"business\_id" string primary key,

"name" string,

"is\_open" number,

"stars" double,

"city" string,

"state" string,

"postal\_code" string,

"checkin\_dates" string

);

insert into "DIM\_BUSINESS"(

"business\_id",

"name",

"is\_open",

"stars",

"city",

"state",

"postal\_code",

"checkin\_dates"

)

select

b."business\_id",

b."name",

b."is\_open",

b."stars",

g."city",

g."state",

g."postal\_code",

ch."date"

from data\_design.ODS."BUSINESS" as b

join data\_design.ODS."GEOGRAPHY" as g on b."geography\_id" = g."geography\_id"

join data\_design.ODS."CHECKIN" as ch on b."business\_id" = ch."business\_id";

-- table fact

create or replace table "FACT"(

"fact\_id" string primary key,

"business\_id" string references data\_design.DWH."DIM\_BUSINESS"("business\_id"),

"customer\_id" string references data\_design.DWH."DIM\_CUSTOMER"("customer\_id"),

"date" date references data\_design.DWH."DIM\_TEMPERATURE"("date" ),

"stars" double

);

insert into "FACT"

select

r."review\_id",

r."business\_id",

r."user\_id",

r."date",

r."stars"

from

data\_design.ODS."REVIEW" as r;