Analyzing the Impact of Business Hour Mismatch on Order Volume in the Food Delivery Industry: A Case Study of UEats and Ghub

# Step 1: Extract Business Hours from UberEats

Since UberEats data is stored in a nested JSON format, we need to extract the startTime and endTime dynamically using BigQuery UDF (User Defined Function).

-- Create a temporary function to extract values dynamically

CREATE TEMP FUNCTION CUSTOM\_JSON\_EXTRACT(json STRING, json\_path STRING)

RETURNS STRING

LANGUAGE js AS """

try {

let obj = JSON.parse(json);

let pathParts = json\_path.split('.');

let result = obj;

for (let part of pathParts) {

if (part === '\*') {

part = Object.keys(result)[0]; // Handle dynamic keys

}

if (result[part] !== undefined) {

result = result[part];

} else {

return null;

}

}

return typeof result === 'object' ? JSON.stringify(result) : result.toString();

} catch (e) {

return null;

}

""";

-- Extract business hours from UberEats data

SELECT

b\_name AS business\_name,

vb\_name AS virtual\_brand,

slug AS ubereats\_slug,

timestamp,

CUSTOM\_JSON\_EXTRACT(TO\_JSON\_STRING(response), 'data.menus.\*.sections.0.regularHours.0.startTime') AS ubereats\_start\_time,

CUSTOM\_JSON\_EXTRACT(TO\_JSON\_STRING(response), 'data.menus.\*.sections.0.regularHours.0.endTime') AS ubereats\_end\_time

FROM

`arboreal-vision-339901.take\_home\_v2.virtual\_kitchen\_ubereats\_hours`

LIMIT 1000;

# Step 2: Extract Business Hours from Grubhub

Unlike UberEats, Grubhub data uses a **different JSON structure**. We extract business hours using **JSON\_EXTRACT\_SCALAR**:

-- Extract business hours from Grubhub data

SELECT

b\_name AS business\_name,

vb\_name AS virtual\_brand,

slug AS grubhub\_slug,

timestamp,

JSON\_EXTRACT\_SCALAR(response, '$.availability\_by\_catalog.STANDARD\_DELIVERY.schedule\_rules[0].from') AS grubhub\_start\_time,

JSON\_EXTRACT\_SCALAR(response, '$.availability\_by\_catalog.STANDARD\_DELIVERY.schedule\_rules[0].to') AS grubhub\_end\_time

FROM `arboreal-vision-339901.take\_home\_v2.virtual\_kitchen\_grubhub\_hours`

LIMIT 1000;

# Step 3: Compute Business Hours Mismatch

Now, we join both datasets to compare their **start time** and **end time**.

WITH UberEats\_Hours AS (

SELECT

slug AS ue\_slug,

b\_name AS ue\_business\_name,

vb\_name AS ue\_virtual\_brand,

timestamp AS ue\_timestamp,

start\_time AS ue\_start\_time,

end\_time AS ue\_end\_time

FROM sunil.ubereats

),

Grubhub\_Hours AS (

SELECT

slug AS gh\_slug,

b\_name AS gh\_business\_name,

vb\_name AS gh\_virtual\_brand,

timestamp AS gh\_timestamp,

start\_time AS gh\_start\_time,

end\_time AS gh\_end\_time

FROM sunil.grubhub

)

SELECT

gh.gh\_slug AS grubhub\_slug,

gh.gh\_business\_name AS grubhub\_name,

gh.gh\_virtual\_brand AS grubhub\_virtual\_brand,

gh.gh\_start\_time AS grubhub\_start\_time,

gh.gh\_end\_time AS grubhub\_end\_time,

ue.ue\_slug AS ubereats\_slug,

ue.ue\_business\_name AS ubereats\_name,

ue.ue\_virtual\_brand AS ubereats\_virtual\_brand,

ue.ue\_start\_time AS ubereats\_start\_time,

ue.ue\_end\_time AS ubereats\_end\_time,

CASE

WHEN gh.gh\_start\_time = ue.ue\_start\_time

AND gh.gh\_end\_time = ue.ue\_end\_time THEN 'In Range'

WHEN ABS(TIMESTAMPDIFF(MINUTE, gh.gh\_start\_time, ue.ue\_start\_time)) <= 5

AND ABS(TIMESTAMPDIFF(MINUTE, gh.gh\_end\_time, ue.ue\_end\_time)) <= 5 THEN 'Out of Range with 5 mins difference'

ELSE 'Out of Range'

END AS is\_out\_range

FROM Grubhub\_Hours gh

JOIN UberEats\_Hours ue

ON gh.gh\_slug = ue.ue\_slug;

# Step 4: Expected Output

