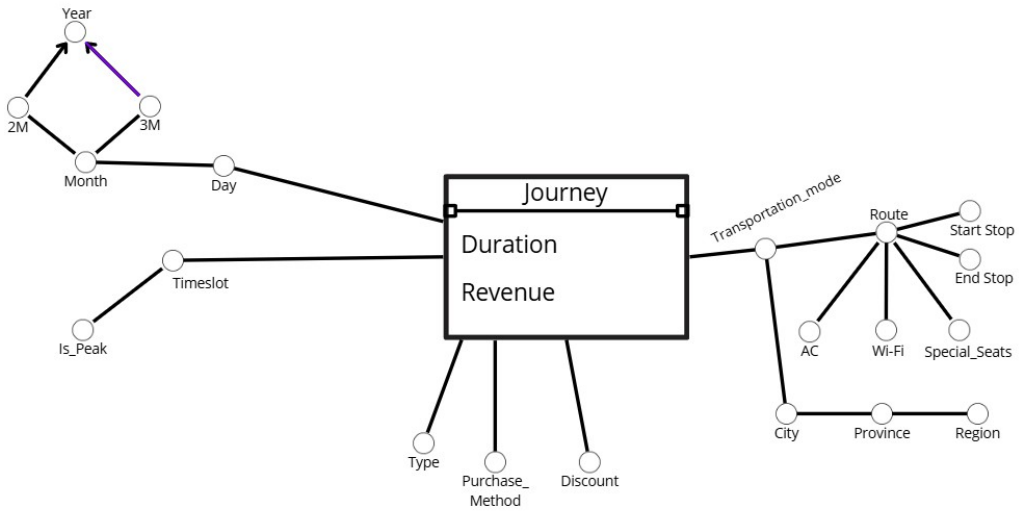

Logical Schema:

TIMESLOT(TSID, Is_Peak)

TIME(TID, Day, Month, 2M, 3M, Year)

TRANSPORTATION(TRID, Transporation_Mode, RID, City, Province, Region, AC, Wi-Fi, Special_Seats, Start_Stop, End_Stop)

JOURNEY(TSID, TID, TRID, JID, Duration, Revenue, Type, Purchase_method, Discount)

Task 2 Extended SQL Queries

a. 1 journey = 1 ticket

```
SELECT TR.Transportation_Mode, T.Year, T.Month, COUNT(JID) / COUNT(DISTINCT Day) AS AvgNumTickets,
SUM(COUNT(JID)) OVER (PARTITION BY Year ORDER BY Month ROWS UNBOUNDED PRECEDING) AS
TrailingYearlyNumTickets,
COUNT(JID) * 100 / SUM(COUNT(JID)) OVER (PARTITION BY Year, Month) AS PercentageTickets
FROM TRANSPORTATION TR, TIME T, JOURNEY J
WHERE J.TID = T.TID AND J.TRID = TR.TRID
GROUP BY TR.Transportation_Mode, T.Year, T.Month
```

b rifatta

```
SELECT SUM(SUM(J.Duration)) OVER (PARTITION BY Transport_Mode, City) / COUNT(JID) AS AvgDuration,
SUM(SUM(Revenue)) OVER (PARTITION BY City) AS RevByCity,
SUM(Revenue) * 100.0 / SUM(SUM(Revenue)) OVER (PARTITION BY Transport_Mode, City) AS
PercentageOfRevByTransportMode,
RANK() OVER (PARTITION BY Transportation_Mode, Start_Stop, End_Stop ORDER BY SUM(Revenue), DESC)
AS RevRanking
```

FROM JOURNEY J, TRANSPORTATION TR, TIME T
WHERE J.TID = T.TID AND J.TRID = TR.TRID
AND Year = 2022
GROUP BY Transportation_Mode, City, Start_Stop, End_Stop

3)

A)Separately for each transportation mode and for each month, analyze the average daily number of tickets.
SELECT Transportation_Mode, Year, Month, COUNT(JID) / COUNT(DISTINCT Day)
FROM JOURNEY J, TIME T, TRANSPORTATION TR
WHERE J.TID = T.TID
AND TR.TRID = J.TRID
GROUP BY Transportation_Mode, Year, Month

GROUP BY: Transportation_Mode, Month
WHERE:
Measures: COUNT(JID), COUNT(Distinct Day)
FROM: JOURNEY, TIME, TRANSPORTATION

B)Separately for each transportation mode and for each month, analyze the cumulative number of tickets from the beginning of the year.
SELECT Transportation_Mode, Year, Month, SUM(COUNT(JID)) OVER (PARTITION BY Year ORDER BY Month ROWS UNBOUNDED PRECEDING) AS TrailingYearlyNumTickets
FROM JOURNEY J, TIME T, TRANSPORTATION TR
WHERE J.TID = TR.TID AND J.TRID = TR.TRID
GROUP BY Transportation_Mode, Year, Month

GROUP BY: Transportation_Mode, Year, Month
WHERE:
Measures: SUM(COUNT(JID))
FROM: JOURNEY, TIME, TRANSPORTATION

C)Separately for each transportation mode and for each month, analyze the total number of tickets sold, the total revenue, and the average revenue.
SELECT Transportation_Mode, Year, Month, COUNT(JID), AVG(Revenue), SUM(Revenue)
FROM JOURNEY J, TRANSPORTATION TR, TIME T
WHERE J.TID = T.TID AND J.TRID = TR.TRID
GROUP BY Transportation_Mode, Year, Month

GROUP BY:Transportation_Mode, Year, Month
WHERE:
Measures:COUNT(JID), SUM(Revenue) / COUNT(JID), SUM(Revenue)
FROM JOURNEY, TRANSPORTATION, TIME

D)Separately for each transportation mode and for each month, analyze the total number of tickets sold, the total revenue, and the average revenue for the year 2024.

```
SELECT Transportation_Mode, Year, Month, COUNT(JID), SUM(Revenue) / COUNT(JID), SUM(Revenue)
FROM JOURNEY J, TRANSPORTATION TR, TIME T
WHERE J.TID = T.TID AND J.TRID = TR.TRID
AND T.Year = 2024
GROUP BY  Transportation_Mode, Year, Month
```

GROUP BY:Transportation_Mode, Month
WHERE: Year
Measures:COUNT(JID), SUM(Revenue)
FROM: JOURNEY, TRANSPORTATION, TIME

E)Analyze the percentage of tickets related to each transportation mode and month over the total number of tickets of the month for each transportation mode.

```
SELECT COUNT(JID) * 100.0 / SUM(COUNT(JID))
OVER (PARTITION BY Year, Month)
FROM JOURNEY J, TIME T, TRANSPORTATION TR
WHERE J.TRID = TR.TRID AND J.TID = T.TID
GROUP BY Transportation_Mode, Year, Month
```

GROUP BY: Transporation_Mode, Year, Month
WHERE:
Measures: COUNT(JID), SUM(COUNT(JID))
FROM JOURNEY, TRANSPORTATION, TIME

```
CREATE MATERIALIZED VIEW MV1
BUILD IMMEDIATE
REFRESH FAST ON COMMIT
AS
(SELECT Transportation_Mode, Year, Month, COUNT(JID) AS SoldTickets, COUNT(Distinct Day) AS
DistinctDays, SUM(Revenue) AS TotRevenue
FROM JOURNEY J, TRANSPORTATION TR, TIME T
WHERE J.TRID = TR.TRID AND J.TID = T.TID
GROUP BY  Transportation_Mode, Year, Month)
```

```
CREATE MATERIALIZED VIEW LOG ON JOURNEY
WITH SEQUENCE, ROWID
(JID, Revenue)
INCLUDING NEW VALUES;
```

```
CREATE MATERIALIZED VIEW LOG ON TIME
WITH SEQUENCE, ROWID
```

(TID, Day, Year, Month)
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON TRANSPORTATION
WITH SEQUENCE, ROWID
(TRID, Transportation_Mode)
INCLUDING NEW VALUES;

Operations for Table JOURNEY influencing the Materialized View:

- INSERT: COUNT(JID) will be increased, SUM(Revenue) will be increased, AVG(Revenue) will be potentially altered.
- DELETE: COUNT(JID) will be decreased, SUM(Revenue) will be decreased, AVG(Revenue) will be potentially altered.
- UPDATE: all the measures would be potentially altered some way or another.

Operations for Table TRANSPORTATION influencing the Materialized View:

- INSERT: adding a new TRID/Transport_Mode alters the grouping of the MW.
- DELETE: deleting a TRID/Transport_mode removes a group from the MW and from the output of the queries.
- UPDATE: it changes the grouping of the MW (ex the names of the groups)

Operations for Table TIME influencing the Materialized View:

- INSERT: Adding a new TID for a specific Day, Month, Year alters the COUNT(Distinct Day).
- DELETE: removing a TID for a specific Day, Month, YearDays alters the COUNT(Distinct Day).
- UPDATE: changing Days values alters COUNT(Distinct Day)

4)
CREATE TABLE MV1 (
Transportation_Mode VARCHAR(15) CHECK (Transportation_Mode IS NOT NULL),
Year NUMBER CHECK (Year IS NOT NULL),
Month NUMBER CHECK (Month IS NOT NULL),
SoldTickets NUMBER CHECK (SoldTickets IS NOT NULL),
DistinctDays NUMBER CHECK (DistincDays IS NOT NULL)
TotRevenue NUMBER CHECK (TotRevenue IS NOT NULL));

INSERT INTO MV1 (Transportation_Mode, Year, Month, SoldTickets, DistinctDays, TotRevenue)
(SELECT TR.Transportation_Mode, T.Year, T.Month, COUNT(J.JID) AS SoldTickets, COUNT(DISTINCT T.Day)

```
AS DistinctDays, SUM(J.Revenue) AS TotRevenue
FROM JOURNEY J, TIME T, TRANSPORTATION TR
WHERE J.TID = T.TID AND J.TRID = TR.TRID
GROUP BY TR.Transportation_Mode, T.Year, T.Month);
```

Trigger to update changes after an insertion on the JOURNEY table:

```
CREATE OR REPLACE TRIGGER UpdateMW1Trigger
AFTER INSERT ON JOURNEY
FOR EACH ROW
DECLARE
BEGIN
N NUMBER;
V_DistinctDays NUMBER;
V_T_Mode VARCHAR(15);
V_Year NUMBER;
V_Month NUMBER;
BEGIN

SELECT Transportation_Mode
INTO V_T_Mode
FROM TRANSPORTATION TR
WHERE :NEW.TRID = TR.TRID;

SELECT T.Year, T.Month,
INTO V_Year, V_Month
FROM TIME T
WHERE :NEW.TID = T.TID;

SELECT COUNT(*)
INTO N
FROM MV1
WHERE Transportation_Mode = V_T_Mode AND Year = V_Year AND Month = V_Month;

IF (N>0) THEN
```

```
        SELECT COUNT(DISTINCT Day)
        INTO V_DistinctDays
        FROM JOURNEY J, TIME T, TRANSPORTATION TR
        WHERE J.TID = T.TID AND J.TRID = TR.TRID
        AND T.Month = V_Month AND T.Year = V_Year
        AND Transportation_Mode = V_T_Mode;

        UPDATE MV1
        SET     SoldTickets = SoldTickets + 1,
                TotRevenue = TotRevenue + :NEW.Revenue,
                DistinctDays =   V_DistinctDays
        WHERE Transportation_Mode = V_T_Mode AND Year = V_T_Year AND Month = V_T_Month
ELSE
        INSERT INTO MV1(Transportation_Mode, Year, Month, SoldTickets, DistinctDays, TotRevenue)
        VALUES (V_T_Mode, V_Year, V_Month, 1, 1, :NEW.Revenue);
END IF;
END
```