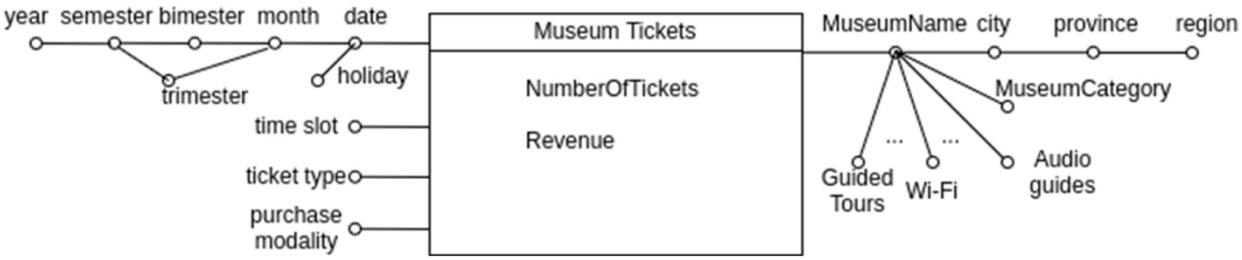
Draft solution



Museum (<u>Id_museum</u>, museum_name, city, province, region, museum_category, guided_tour, wifi,.., audioguides)
TimeDim (<u>id_time</u>, date, month, bimester, trimester, semester, year, holiday)
MuseumTickets (<u>Id_museum</u>, <u>id_time</u>, <u>timeslot</u>, <u>ticket_type</u>, <u>purchase_modality</u>, NumberOfTickets, Revenue)

Queries

Separately for each ticket type and for each month (of the ticket validity), analyze: the average daily revenue, the cumulative revenue from the beginning of the year, the percentage of tickets related to the considered ticket type over the total number of tickets of the month.

```
SELECT ticket_type, month,
      sum(revenue)/count(distinct date),
      sum(sum(revenue)) over (partition by ticket_type, year
                              order by month rows unbounded preceding),
      100*sum(numtickets)/sum(sum(numtickets)) over (partition by month)
FROM museums_tickets mt, timedim t
WHERE mt.id_time = t.id_time
GROUP BY ticket_type, month, year;
```

Considering the ticket of 2021. Separately for each museum and ticket type analyze: the average revenue for a ticket, the percentage of revenue over the total revenue for the corresponding museum category, assign a rank to the museum, for each ticket type, according to the total number of tickets in decreasing order.

```
SELECT museum_name, museum_category, ticket_type,
      sum(revenue)/sum(numtickets),
      100*sum(revenue)/sum(sum(revenue)) over (partition by ticket_type,
         museum_category),
      rank() over (partition by ticket_type order by sum(num_tickets)
      desc)
FROM museums_tickets mt, timedim t, museums m
```

```
WHERE mt.id_time = t.id_time and mt.id_museum = m.id_museum and year=2021
GROUP BY id_museum, museum_name, museum_category, ticket_type;
```

Create and update a materialized view with CREATE MATERIALIZED VIEW and CREATE MATERIALIZED VIEW LOG in ORACLE.

```
create materialized view GROUPBYMonthYearMuseum
build immediate
refresh FAST ON COMMIT
--enable query rewrite
as
SELECT Month, Year, ticket_type , SUM(num_tickets) as NumTickets,
       SUM(revenue) as TotRevenue
FROM museums_tickets mt, timedim t
WHERE mt.id_time = t.id_time
GROUP BY Month, Year, ticket_type;

CREATE MATERIALIZED VIEW LOG ON museums_tickets
WITH SEQUENCE, ROWID (id_time, ticket_type, num_tickets, Revenue)
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON TIMEDIM
WITH SEQUENCE, ROWID (id_time, Month, Year)
INCLUDING NEW VALUES;
```

Update and management of views via Trigger

```
CREATE TABLE VM1 (
      DateMonth DATE NOT NULL),
      DateYear INTEGER NOT NULL,
      Ticket_Type VARCHAR(20) NOT NULL,
      TOT_NumberOfTickets INTEGER,
      TOT_Revenue INTEGER);

INSERT INTO VM1 (DateMonth, DateYear, Ticket_Type, TOT_NumberOfTickets,
TOT_Revenue)
(SELECT Month, Year, Ticket_Type,
SUM(NumberOfTickets), SUM(Revenue)
FROM museums_tickets mt, timedim t
WHERE mt.id_time = t.id_time
GROUP BY Month, Year, Ticket_Type);


create TRIGGER TriggerForViewVM1
AFTER INSERT ON museums_tickets
FOR EACH ROW
DECLARE
      N NUMBER;
      VAR_DateMonth DATE;
      VAR_DateYear NUMBER;
```

```
BEGIN
      SELECT Month, Year INTO VAR_DateMonth, Var_DateYear
      FROM TIMEDIM
      WHERE ID_time = :NEW.ID_time;

      SELECT Count(*) INTO N
      FROM VM1
      WHERE DateMonth=Var_DateMonth AND DateYear = Var_DateYear and
      Ticket_Type=:NEW.ticket_type;

      if (N > 0) then

            update VM1
            set TOT_NumberOfTickets = TOT_NumberOfTickets +
            :NEW.NumberOfTickets, TOT_Revenue = TOT_Revenue + :NEW.Revenue
            where DateMonth= Var_DateMonth AND DateYear = Var_DateYear
            and Ticket_Type=:NEW.ticket_type;
      else

            insert into VM1 (DateMonth, DateYear, Ticket_Type,
            Tot_NumberOfTickets, Tot_Revenue)
            values (Var_DateMonth, Var_DateYear, :NEW.ticket_type,
            :NEW.NumberOfTickets, :NEW.Revenue);

      end if;
END;
```