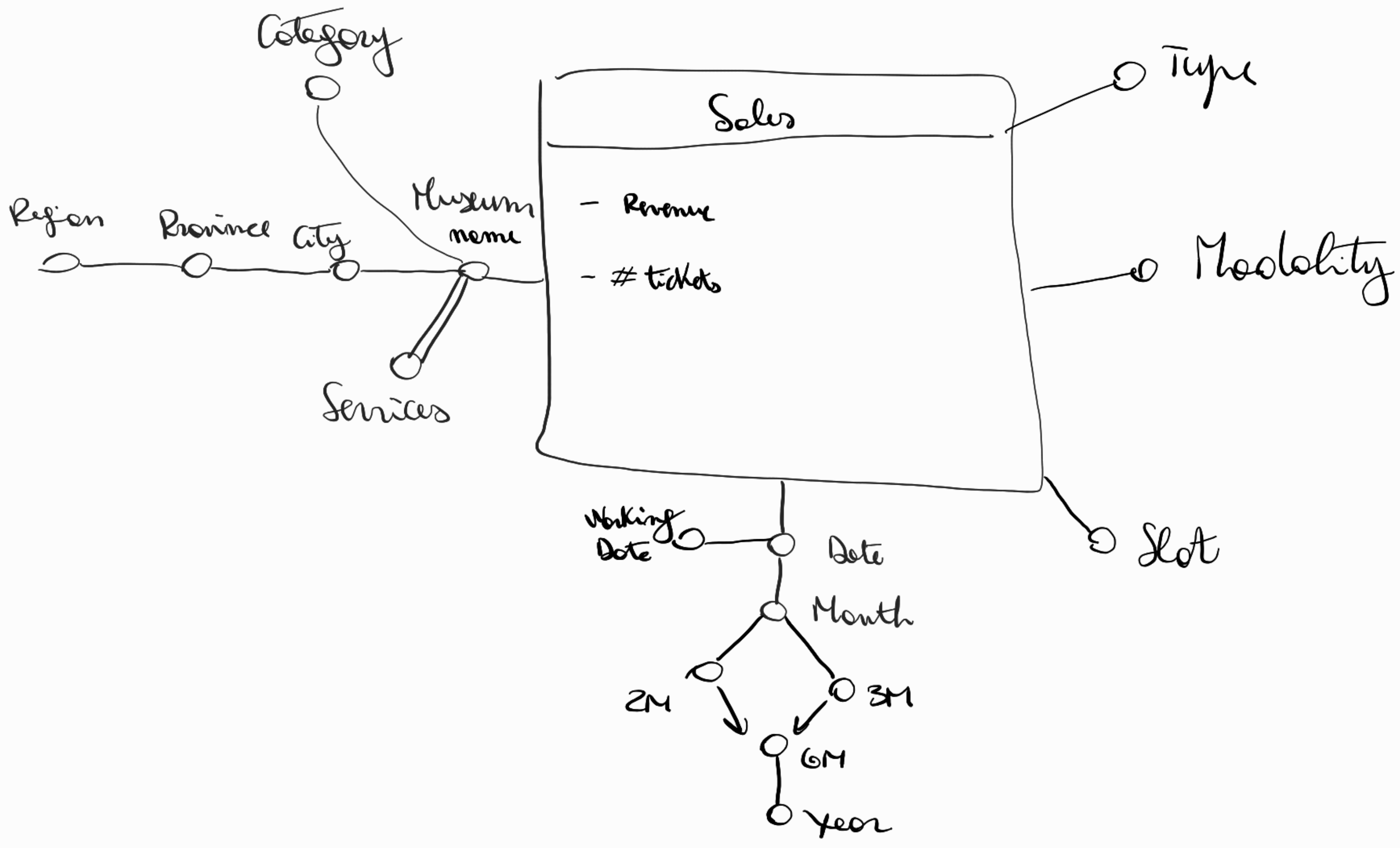
Exercise 1



MUSEUM (<u>MID</u>, MuseumName, Category, City, Province, Region)
SERVICES_OF_MUSEUMS (<u>SID</u>, <u>MID</u>)
SERVICE (<u>SeID</u>, ServiceName)
TIME (<u>TID</u>, Date, WorkingDate, Month, 2M, 3M, 6M, Year)
SALES (<u>SID</u>, <u>JID</u>, <u>MID</u>, <u>TID</u>, Revenue, #tickets)
JUNK (<u>JID</u>, Type, Modality, Slot)

Exercise 2

```
1) SELECT Type, Month, SUM(Revenue) / COUNT(DISTINCT Date) AS Avg_daily_rev,
SUM(SUM(Revenue)) OVER (PARTITION BY Year, Type ORDER BY Month, ROWS UNBOUNDED PRECEDING) AS
Cumulative_rev,
100 * SUM(#tickets) / SUM(SUM(#tickets)) OVER (PARTITION BY Month) AS Perc_type
FROM TIME T, SALES S, JUNK J
WHERE T.TID = S.TID AND J.JID=S.JID
GROUP BY Type, Month, Year
```

```
2) SELECT MID, Type, SUM(Revenue) / SUM(#tickets) AS Avg_ticket_rev,
100 * SUM (Revenue) / SUM(SUM(Revenue)) OVER (PARTITION BY Type, Category) AS Perc_museum_type,
RANK() OVER (PARTITION BY Type ORDER BY SUM(#tickets) DESC) AS Rank
FROM SALES S, MUSEUM M, TIME T, JUNK J
WHERE S.TID = T.TID AND M.MID = S.MID AND Year = '2021' AND J.JID=S.JID
GROUP BY Type, MID, Category
```

Exercise 3

```
a) SELECT SUM(#ticket) / 6 AS Avg_monthly_rev
FROM TIME T, SALES S, JUNK J
WHERE T.TID = S.TID AND J.JID=S.JID
GROUP BY 6M, Type
```

```
b) SELECT SUM(SUM(Revenue)) OVER (PARTITION BY Year,Type ORDER BY Month ROWS UNBOUNDED PRECEDING)
FROM TIME T, SALES S, JUNK J
WHERE T.TID = S.SID AND J.JID=S.JID
GROUP BY Type, Month, Year
```

```
c) SELECT SUM(#tickets), SUM(Revenue),  SUM(Revenue) / SUM(#tickets)
FROM TIME T, SALES S, JUNK J
WHERE Modality = 'Online' AND T.TID = S.TID AND J.JID=S.JID
GROUP BY Type, Month, Year
```

```
d) SELECT SUM(#tickets), SUM(Revenue), SUM(Revenue) / SUM(#tickets)
FROM TIME T, SALES S, JUNK J
WHERE Year = '2021' AND T.TID = S.TID AND J.JID=S.JID
GROUP BY Month, Type
```

```
e) SELECT 100 * SUM(#tickets) / SUM(SUM(#tickets)) OVER (PARTITION BY Month)
FROM TIME T, SALES S, JUNK J
WHERE T.TID = S.TID AND J.JID=S.JID
GROUP BY Type, Month
```

1) The query that has to be used to create the materialized view is:

```
SELECT Type, Month, 6M, Year, Modality, SUM(#tickets) AS TotalTickets,
SUM(Revenue) AS TotalRevenue
FROM TIME T, SALES S, JUNK J
WHERE T.TID = S.TID AND J.JID=S.JID
GROUP BY Type, Month, 6M, Year, Modality
```

The materialized view is:

```
CREATE MATERIALIZED VIEW VM1
BUILD IMMEDIATE
REFRESH FAST ON COMMIT
ENABLE QUERY REWRITE
AS (
SELECT Type, Month, 6M, Year, Modality, SUM(#tickets) AS TotalTickets,
SUM(Revenue) AS TotalRevenue
FROM TIME T, SALES S, JUNK J
WHERE T.TID = S.TID AND J.JID=S.JID
GROUP BY Type, Month, 6M, Year, Modality
)
```

2)
```
CREATE MATERIALIZED VIEW LOG ON TIME
WITH SEQUENCE, ROWID
(TID, Month, 6M, Year)
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON SALES
WITH SEQUENCE, ROWID
(SID, JID, MID, TID, Revenue, #tickets)
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON JUNK
WITH SEQUENCE, ROWID
(JID, Type, Modality, Slot)
INCLUDING NEW VALUES;
```

3)The operations that cause an update of the materialized view VM1 are:
INSERT INTO SALES
DELETE FROM SALES
UPDATE SALES, TIME, JUNK

Exercise 4

1)
```
CREATE TABLE VM1 (
        Type VARCHAR (20) NOT NULL,
        Month VARCHAR (20) NOT NULL,
        6M VARCHAR (20) NOT NULL,
        Year INTEGER NOT NULL,
        Modality VARCHAR (20) NOT NULL,
        TotalTickets INTEGER NOT NULL,
        TotalRevenue FLOAT NOT NULL,
        PRIMARY KEY(Type, Modality, Month)
);
```

```
2)INSERT INTO VM1 (Type, Modality, Month, 6M, Year, TotalRevenue, TotalTickets)
        (
        SELECT Type, Month, 6M, Year, Modality, SUM(#tickets) AS TotalTickets,
        SUM(Revenue) AS TotalRevenue
        FROM TIME T, SALES S, JUNK J
        WHERE T.TID = S.TID AND J.JID=S.JID
        GROUP BY Type, Month, 6M, Year, Modality
        );
```

```
3)CREATE TRIGGER RefreshViewSales
AFTER INSERT ON SALES
FOR EACH ROW
DECLARE
N, VarYear number;
VarType, VarModality, VarMonth, Var6M VARCHAR(20);

BEGIN

SELECT Month, 6M, Year INTO VarMonth, Var6M, VarYear
FROM TIME
WHERE TID=:NEW.TID;

SELECT Type, Modality INTO VarType, VarModality
FROM JUNK
WHERE JID=:NEW.JID;

SELECT COUNT(*) INTO N
FROM VM1
WHERE Type = VarType AND Modality = VarModality AND Month = VarMonth;

IF (N >0) THEN
        UPDATE VM1
        SET TotalRevenue = TotalRevenue + :New.Revenue,
        TotalTickets = TotalTickets + :New.#tickets
        WHERE Type = VarType AND Modality = VarModality AND Month = VarMonth;

ELSE
        INSERT INTO VM1 (Type, Modality, Month, 6M, Year, TotalRevenue, TotalTickets)
                VALUES (VarType, VarModality, VarMonth, Var6M, VarYear, :NEW.VarTotalRevenue, :NEW.VarTotalTickets);

END IF;
END;
```

4) For example, an operation that activate the trigger is an INSERT on SALES