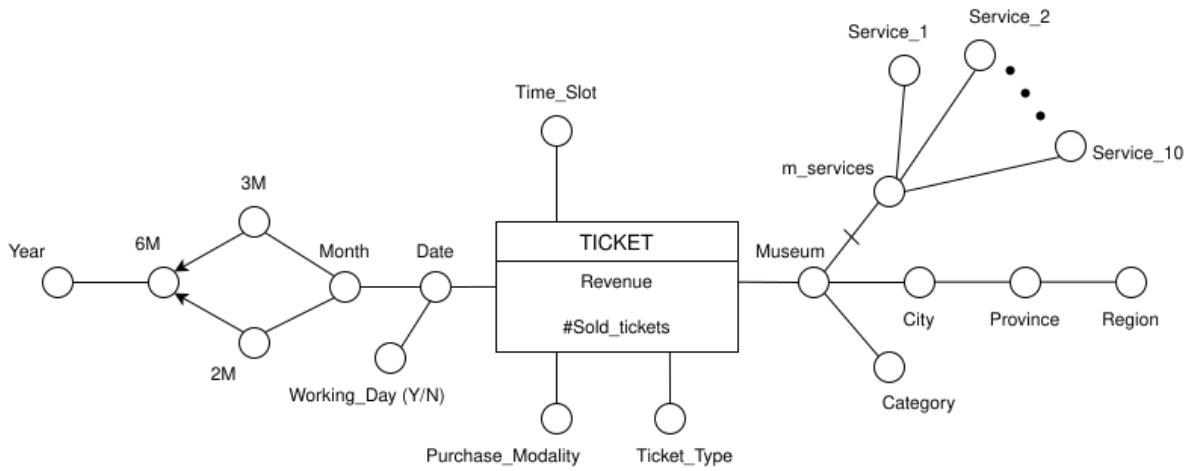
Homework 1

Gabriele Lorenzo
s314913

November 2022

1: Conceptual Schema and Logical Schema

Conceptual Schema



Logical Schema

TIME (<u>TID</u>, Date, Working_Day, Month, 2M, 3M, 6M, Year)

MUSEUM (<u>MID</u>, Museum, Category, City, Province, Region, Service1, Service2, ..., Service10)

TICKET (<u>TKID</u>, <u>TID</u>, <u>MID</u>, <u>Ticket_Type</u>, <u>Purchase_Modality</u>, <u>Time_Slot</u>, Revenue, #Sold_Ticket)

2: Queries

2.1

```
1    SELECT TICKET_TYPE,
2           MONTH,
3           SUM(REVENUE)/COUNT(DISTINCT DATE),
4           SUM(SUM(REVENUE)) OVER (PARTITION BY YEAR
5                                   ORDER BY MONTH
6                                   ROWS UNBOUNDED PRECEDING),
```

```
7           (SUM(#SOLD_TICKETS)/SUM(SUM(#SOLD_TICKETS))) * 100
               OVER (PARTITION BY MONTH)
8   FROM TICKET TK, TIME T
9   WHERE TK.TID = T.TID
10  GROUP BY TICKET_TYPE, MONTH, YEAR;
```

2.2

```
1   SELECT MUSEUM,
2          TICKET_TYPE,
3          SUM(REVENUE)/SUM(#SOLD_TICKETS),
4          (SUM(REVENUE)/SUM(SUM(REVENUE))) * 100 OVER (PARTITION
              BY CATEGORY),
5          RANK() OVER (PARTITION BY TICKET_TYPE, MUSEUM
6                       ORDER BY SUM(#SOLD_TICKETS) DESC)
7   FROM TICKET TK, TIME T, MUSEUM M
8   WHERE TK.TID = T.TID
9       AND TK.MID = M.MID
10      AND YEAR = 2021
11  GROUP BY MUSEUM, TICKET_TYPE;
```

3: Materialized Views

3.1

```
1   CREATE MATERIALIZED VIEW ViewTicket
2   BUILD IMMEDIATE
3   REFRESH FAST ON COMMIT
4   AS
5       SELECT TICKET_TYPE,
6              DATE,
7              MONTH,
8              YEAR,
9              SUM(REVENUE) AS TOT_REVENUE,
10             SUM(#SOLD_TICKETS) AS TOT_SOLD_TICKETS
11      FROM TICKET TK, TIME T
12      WHERE TK.TID = T.TID
13      GROUP BY TICKET_TYPE, MONTH, YEAR, DATE;
```

3.2

```
CREATE MATERIALIZED VIEW LOG ON TICKET
WITH SEQUENCE, ROWID (TKID, TID, MID, TICKET_TYPE,
    PURCHASE_MODALITY, TIME_SLOT, REVENUE, #SOLD_TICKETS)
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON TIME
WITH SEQUENCE, ROWID (TID, DATE, MONTH, YEAR)
INCLUDING NEW VALUES;
```

I don't need to create a Materialized View Log on the Museum Table.

3.3

An **INSERT** operation on tables **TICKET** or **TIME** cause an update of the Materialized View.

4: Triggers

4.1

```
CREATE TABLE VM1 (
    TICKET_TYPE VARCHAR(20),
    DATE DATE,
    MONTH VARCHAR(20),
    YEAR INTEGER,
    TOT_REVENUE INTEGER CHECK (TOT_REVENUE IS NOT NULL AND
        TOT_REVENUE > 0),
    TOT_SOLD_TICKETS INTEGER CHECK (TOT_SOLD_TICKETS IS NOT
        NULL AND TOT_SOLD_TICKETS > 0)
    PRIMARY KEY(TICKET_TYPE, DATE)
);
```

4.2

```
INSERT INTO VM1(TICKET_TYPE, DATE, MONTH, YEAR, TOT_REVENUE,
    TOT_SOLD_TICKETS)
(
```

```
    SELECT TICKET_TYPE,
           DATE,
           MONTH,
           YEAR,
           SUM(TOT_REVENUE),
           SUM(SOLD_TICKETS)
    FROM TICKET TK, TIME T
    WHERE TK.TID = T.TID
    GROUP BY TICKET_TYPE, DATE, MONTH, YEAR;
);
```

4.3

```
CREATE TRIGGER RefreshViewTicket
AFTER INSERT ON TICKET
FOR EACH ROW

DECLARE

varDate DATE;
varMonth VARCHAR(20);
varYear INTEGER;
N INTEGER;

BEGIN

SELECT DATE, MONTH, YEAR INTO varDate, varMonth, varYear
FROM TIME
WHERE TID = :NEW.TID;

SELECT COUNT(*) INTO N
FROM ViewTicket
WHERE DATE = varDate
    AND TICKET_TYPE = :NEW.TICKET_TYPE

IF (N>0) THEN
UPDATE VM1
SET TOT_REVENUE = TOT_REVENUE + :NEW.REVENUE,
    TOT_SOLD_TICKETS = TOT_SOLD_TICKETS + :NEW.#SOLD_TICKETS
WHERE DATE = varDate
```

```
28          AND TICKET_TYPE = :NEW.TICKET_TYPE
29      ELSE
30          INSERT INTO VM1 (TICKET_TYPE, DATE, MONTH, YEAR,
                TOT_REVENUE, TOT_SOLD_TICKETS)
31          VALUES (:NEW.TICKET_TYPE, varDate, varMonth, varYear,
                :NEW.REVENUE, :NEW.#SOLD_TICKETS);
32      END IF;
33      END;
```

4.4

An **INSERT** operation on table **TICKET** triggers the trigger created in 4.3.