Data science and database technology

Exercise on materialized views: materialized view update using triggers

The following relational schema of a data mart is given

INCOME (BranchID, ServiceID, CompanyID, TimeID, #Consultancies, Income)
SERVICE (ServiceID, Consulting, Type, Category)
TIME (TimeID, Date, Month, 2M, 3M, 4M, Semester, Year)
CONSULTANTS_BRANCH (BranchID, Branch, City, Region, GeographicArea, #Consultants)
COMPANY (CompanyID, CompanyCategory, CompanyType, Nationality)

The VM1 materialized view is given as defined as follows

- VM1 contains the income cashed and the number of consultancies provided, separately for "Type of Service" provided, "Geographic area" of the consultants' branch and "Semester".

- VM1 is characterized by the following schema

VM1 (ServiceType, GeographicAreaBranch, Semester, TotIncome, TotNumConsultancies)

- VM1 must be managed **without using** the CREATE MATERIALIZED VIEW statement. The derived table corresponding to the materialized view VM1 is defined by the following SQL statement:

```
CREATE TABLE VM1 (ServiceType VARCHAR(20),
                  GeographicAreaBranch VARCHAR(20),
                  Semester VARCHAR(20),
                  TotIncome INTEGER CHECK (TotIncome IS NOT NULL and TotIncome >0),
                  TotNumConsultancies  INTEGER
                  CHECK (TotNumConsultancies IS NOT NULL and TotNumConsultancies >0),
                  PRIMARY KEY (ServiceType, GeographicAreaHeadquarterC, Semester)  )
```

- The following dependencies between attributes are available, but they are not managed by integrity constraints (no foreign key constraints are defined):
  - ServiceType has SERVICE(Type) as a domain
  - GeographicAreaBranch has CONSULTANTS_BRANCH(GeographicArea) as a domain
  - Semester has TIME(Semester) as a domain

Managing updates on the derived table (materialized view) VM1 requires writing specific triggers.

Point 1: Write the INSERT statement for the initial data loading into the VM1 derived table.

Note: Records can be inserted into a relational table using the INSERT (SELECT ...) statement.

```
INSERT INTO VM1 (ServiceType, GeographicAreaBranch, Semester, TotIncome, TotNumConsultancies)
        (SELECT Type, GeographicArea, Semester, SUM(Income), SUM(#Consultancies)
        FROM INCOME I, SERVICE S, TIME T, CONSULTANTS_BRANCH SC
        WHERE S.ServiceID = I.ServiceID AND SC.BranchID=I.BranchID AND T.TimeID=I.TimeID
        GROUP BY Type, GeographicArea, Semester);
```

Point 2: Write the trigger to propagate the value update of the Type of Service attribute in the Service table to the VM1 materialized view. Note: Consider the type 1 mode for time management

```
CREATE TRIGGER UpdateTypeOfService
AFTER UPDATE OF Type ON SERVICE
FOR EACH ROW
DECLARE
 N number;
BEGIN

---check if there is at least one record associated with the previous value of service type
SELECT Count(*) INTO N
FROM VM1
WHERE ServiceType = :OLD.Type;

IF (N > 0) THEN
--- Update the records associated with the previous value of service type
UPDATE VM1
SET ServiceType = :NEW.Type
WHERE ServiceType = :OLD.type;

END IF;
END;
```

Point 3 Write the trigger to propagate to VM1 the changes due to the insertion of a new record in the INCOME fact table.

```
CREATE TRIGGER InsertNewServiceType
AFTER INSERT ON INCOME
FOR EACH ROW
DECLARE
N number;
TypeServiceVar VARCHAR(20); GeographicAreaBranchVar VARCHAR(20); SemesterVar VARCHAR(20);

BEGIN
--- check if there is a record in VM1 to be updated. In this record, the values of attributes ServiceType, GeographicAreaHeadquarterC and Semester, are related to the values of NEW.ServiceId,: NEW.IdSede,: NEW.TimeId in the new record inserted in the INCOME table.

--- read the corresponding values in the different dimensional tables
SELECT Type INTO ServiceTypeVar
FROM SERVICE
WHERE ServiceID = :NEW.ServiceID;

SELECT GeographicArea INTO GeographicAreaBranchVar
FROM CONSULTANTS_BRANCH
WHERE BranchID = :NEW.BranchID;

SELECT Semester INTO SemesterVar
FROM TIME
WHERE TimeID = :NEW.TimeID;


SELECT Count(*) INTO N
FROM VM1
WHERE ServiceType = ServiceTypeVar AND GeographicAreaBranch = GeographicAreaBranchVar
AND Semester = SemesterVar;

if (N > 0) then
--- Update the existing record
        UPDATE VM1
        SET TotIncome = TotIncome + :NEW.Income,
            TotNumConsultancies = TotNumConsultancies + :NEW.#Consultancies
        WHERE ServiceType = ServiceTypeVar
        AND GeographicAreaBrach = GeographicAreaBrachVar
        AND Semester = SemesterVar;
else
--- insert a new record
        INSERT INTO VM1 (ServiceType, GeographicAreaBranch, Semester, TotIncome,
TotNumConsultancies)
        VALUES (ServiceTypeVar, GeographicAreaBranchVar, SemesterVar, :NEW.Income,
:NEW.#Consultancies);
END IF; END;
```

Materialized Views Exercise: Updating materialized view Using CREATE MATERIALIZED VIEW LOG and CREATE MATERIALIZED VIEW in Oracle

The following relational schema of a data mart is given

INCOME (BranchID, ServiceID, CompanyID, TimeID, #Consultancies, Income)
SERVICE (ServiceID, Consulting, Type, Category)
TIME (TimeID, Date, Month, 2M, 3M, 4M, Semester, Year)
CONSULTANTS_BRANCH (BranchID, Branch, City, Region, GeographicArea, #Consultants)
COMPANY (CompanyID, CompanyCategory, CompanyType, Nationality)

The VM1 materialized view is given as defined as follows

- VM1 contains the income cashed and the number of consultancies provided, separately for "Type of Service" provided, "Geographic area" of the consultants' branch and "Semester".

- VM1 is characterized by the following schema

VM1 (ServiceType, GeographicAreaBranch, Semester, TotIncome, TotNumConsultancies)

The VM1 materialized view is created with the following SQL statement

```
CREATE MATERIALIZED VIEW
BUILD IMMEDIATE
REFRESH FAST ON DEMAND
ENABLE QUERY REWRITE
AS      <Query>
```

Point 1: **Define the query needed to complete the definition of materialized view VM1**

```
(SELECT Type As ServiceType, GeographicArea AS GeographicAreaBranch, Semester,
SUM(Income) AS TotIncome, SUM(#Consultancies) AS TotNumConsultancies
FROM INCOME I, SERVICE S, TIME T, CONSULTANTS_BRANCH SC
WHERE S.ServiceID = I.ServiceID AND SC.BranchID = I.BranchID AND T.TimeID = I.TimeID
GROUP BY Type, GeographicArea, Semester);
```

<u>Point 2</u>: Write the instructions that define the MATERIALIZED VIEW LOG in Oracle necessary for the automatic FAST update of the VM1 materialized view. Indicate *all and only* the necessary logs and within each log definition indicate *all and only* the necessary attributes.

```
CREATE MATERIALIZED VIEW LOG ON INCOME
WITH SEQUENCE, ROWID
(BranchID, ServiceID, TimeID, #Consultancies, Income)
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON SERVICE
WITH SEQUENCE, ROWID
(ServiceID, Type)
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON TIME
WITH SEQUENCE, ROWID
(TimeID, Semester)
INCLUDING NEW VALUES;

CREATE MATERIALIZED VIEW LOG ON CONSULTANTS_BRANCH
WITH SEQUENCE, ROWID
(BranchID, GeographicArea)
INCLUDING NEW VALUES;
```