1 View 1:UpcomingByHospital

1.1 Description

This view encapsulates a complex query that joins multiple tables and counts scheduled appointments per hospital and date. By using it, the application can query Upcoming-ByHospital directly without rewriting the joins and conditions each time

1.2 SQL Definition

```
CREATE VIEW UpcomingByHospital AS
SELECT h.Name,c.Date, COUNT(a.CAID)
FROM Appointment as a
JOIN ClinicalActivity c ON a.CAID = c.CAID
JOIN Department d ON c.DEP_ID = d.DEP_ID
JOIN  Hospital h ON d.HID = h.HID
WHERE a.Status='Scheduled'
      AND c.Date BETWEEN CURRENT_DATE AND CURRENT_DATE + INTERVAL 14
         DAY
GROUP BY h.Name , c.Date ;
-- QUERY EXAMPLE FOR VIEW 1

SELECT *
FROM UpcomingByHospital
ORDER BY Name, Date;
```

1.3 Output Example

Figure 1: Output of the view query

2 View 2:DrugPricingSummary

2.1 Description

The view DrugPricingSummary encapsulates the complex joins of several tables to return the summary of medication per hospital .

2.2 SQL Definition

```
CREATE VIEW DrugPricingSummary AS
SELECT H.HID , H.Name AS HospitalName, M.MID , M.Name AS MedicationName
    ,
    MAX(S.UnitPrice)  AS MaxUnitPrice,
```

```
    min(S.UnitPrice)  AS MinUnitPrice,
    AVG(S.UnitPrice) AS AvgUnitPrice,
    MAX(S.StockTimestamp) AS LastStockTimestamp
from stock S
join Hospital H on S.HID = H.HID
join Medication M on S.MID = M.MID
group by H.HID, H.Name, M.MID, M.Name;

-- QUERY EXAMPLE FOR VIEW 2
SELECT * FROM DrugPricingSummary ;
```

2.3 Output Example

129 21:46:44 SELECT * FROM DrugPricingSummary ; INSERT INTO Stock (HID, MID, StockTimestamp, UnitPrice... Error Code: 1644 Insert rejected: Qty cannot be negative

HID	HospitalName	MID	MedicationName	MaxUnitPrice	MinUnitPrice	AvgUnitPrice	LastStockTimestamp
1	Benguerir Central Hospital	1001	Amoxicillin	22.00	22.00	22.000000	2025-10-10 08:00:00
1	Benguerir Central Hospital	1002	Ibuprofen	6.50	6.50	6.500000	2025-10-10 08:00:00
2	Casablanca University Hospital	1003	Azithromycin	35.00	35.00	35.000000	2025-10-12 08:00:00
3	Rabat Clinical Center	1004	Paracetamol	4.00	4.00	4.000000	2025-10-15 08:00:00
4	Marrakech Regional Hospital	1005	Ceftriaxone	120.00	120.00	120.000000	2025-10-20 08:00:00

Figure 2: Output of the view query

3 View 3:StaffWorkloadThirty

3.1 Description

The StaffWorkloadThirty view simplifies application development by encapsulating the complex joins and 30-day date filtering into a single object. we are able now to retrieve the whole staff workload in the last thirty days using a SELECT * query. This ensures data consistency and avoids redundant logic when trying to retrieve this same data without the view.

3.2 SQL Definition

```
CREATE  VIEW StaffWorkloadThirty AS
SELECT
    S.STAFF_ID,
    S.FullName,
    COUNT(A.CAID) AS TotalAppointments,
    COUNT(CASE WHEN A.Status = 'Scheduled' THEN 1 END) AS
        ScheduledCount,
    COUNT(CASE WHEN A.Status = 'Completed' THEN 1 END) AS
        CompletedCount,
    COUNT(CASE WHEN A.Status = 'Cancelled' THEN 1 END) AS
        CancelledCount
FROM
    Staff S
LEFT JOIN
    ClinicalActivity CA ON S.STAFF_ID = CA.STAFF_ID
LEFT JOIN
```

```
    Appointment A
    ON CA.CAID = A.CAID
    AND CA.Date >= CURRENT_DATE() - INTERVAL 30 DAY
GROUP BY
    S.STAFF_ID,
    S.FullName;

-- QUERY EXAMPLE OF VIEW 3

SELECT * FROM StaffWorkloadThirty ;
```

3.3 Output Example

STAFF_ID	FullName	TotalAppointments	ScheduledCount	CompletedCount	CancelledCount
501	Dr. Amina Idrissi	0	0	0	0
502	Dr. Mehdi Touil	0	0	0	0
503	Nurse Firdawse Guerbouzi	0	0	0	0
504	Technician Omar Lahlou	0	0	0	0
505	Dr. Khaoula Messari	0	0	0	0

Figure 3: Output of the view query

4 View 4:PatientNextVisit

4.1 Description

The view PatientNextVisit returns for each patient the next scheduled visit . The view improves data retrieval by allowing the encapsulation of all the joins without redundancy .

4.2 SQL Definition

```
CREATE VIEW PatientNextVisit AS
SELECT p.IID ,
    p.FullName,
    MIN(c.date) as NextAppointment,
    d.Name as DepartmentName,
    h.Name as HospitalName,
    h.City as HospitalCity
FROM Patient as p
JOIN ClinicalActivity c on c.IID = p.IID
JOIN Appointment a on a.CAID = c.CAID
JOIN Department as d on d.DEP_ID = c.DEP_ID
JOIN Hospital as h on h.HID = d.HID
Where a.status = 'Scheduled'
AND c.date > CURRENT_DATE ;
GROUP BY
p.IID,
p.FullName,
d.Name,
```

```
h.Name,
h.City;
```

4.3 Output Example

IID	FullName	NextAppointment	DepartmentName	HospitalName	HospitalCity

Figure 4: Output of the view query

5 Trigger 1: Reject double booking for a staff member

5.1 Trigger Definition

```
DELIMITER ||
CREATE TRIGGER RejectDoubleAppInsert
BEFORE INSERT on Appointment
FOR EACH ROW
BEGIN
    DECLARE count_app INT ;
    SELECT Count(*)
    INTO count_app
    FROM Appointment as a
    JOIN ClinicalActivity as c on c.CAID = a.CAID
    WHERE c.STAFF_ID = ( SELECT STAFF_ID FROM ClinicalActivity WHERE
       CAID = NEW.CAID)
        AND c.Time = ( SELECT Time FROM ClinicalActivity WHERE CAID =
           NEW.CAID)
        AND c.Date = ( SELECT Date FROM ClinicalActivity WHERE CAID =
           NEW.CAID) ;
    IF count_app > 0 THEN
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'The time and date slot chosen are
               already taken';
    END IF ;

END ||


DELIMITER ||
CREATE TRIGGER RejectDoubleAppUpdate
BEFORE UPDATE on Appointment
FOR EACH ROW
BEGIN
    DECLARE count_app INT ;
    SELECT Count(*)
    INTO count_app
    FROM Appointment as a
    JOIN ClinicalActivity as c on c.CAID = a.CAID
    WHERE c.STAFF_ID = ( SELECT STAFF_ID FROM ClinicalActivity WHERE
       CAID = NEW.CAID)
```

```
        AND c.Time = ( SELECT Time FROM ClinicalActivity WHERE CAID =
            NEW.CAID)
        AND c.Date = ( SELECT Date FROM ClinicalActivity WHERE CAID =
            NEW.CAID) ;
    IF count_app > 0 THEN
        SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'The time and date slot chosen are
                already taken';
    END IF ;

END ||


-- QUERY EXAMPLE FOR TRIGGER 1
UPDATE Appointment
SET CAID = 1001
WHERE CAID = 1001;

INSERT INTO ClinicalActivity (CAID, IID, STAFF_ID, DEP_ID, Date, Time)
VALUES (2001, 2, 501, 10, '2025-10-10', '10:00:00');
INSERT INTO Appointment (CAID, Reason)
VALUES (2001, 'Duplicate booking test');
```

5.2 Trigger Output

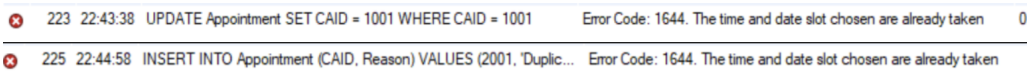

Figure 5: Error message returned by the trigger

6 Trigger 2:Recompute Expense Total when prescription lines change.

6.1 Trigger Definition

6.1.1 missingPrice Trigger

```
DELIMITER ||

CREATE TRIGGER missingprice
BEFORE INSERT ON Includes
FOR EACH ROW
BEGIN
    DECLARE v_HID INT;
    DECLARE v_UnitPrice DECIMAL(10, 2);

    SELECT H.HID
    INTO v_HID
    FROM Prescription P
    JOIN ClinicalActivity CA ON P.CAID = CA.CAID
    JOIN Department D ON CA.DEP_ID = D.DEP_ID
```

```
    JOIN Hospital H ON D.HID = H.HID
    WHERE P.PID = NEW.PID;

    SELECT S.UnitPrice
    INTO v_UnitPrice
    FROM Stock S
    JOIN Medication M ON S.MID = M.MID
    JOIN Hospital H ON S.HID = H.HID
    WHERE S.MID = NEW.MID
      AND S.HID = v_HID
    ORDER BY S.StockTimestamp DESC
    LIMIT 1;

    IF v_UnitPrice IS NULL THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'No valid Stock UnitPrice found for this
           medication at the hospital location.';
    END IF;

END ||

-- missingprice test

INSERT INTO Includes(PID, MID, Dosage, Duration)
VALUES (8001, 1004, '1 tab OD', '3 days');
```

401 12:27:37 INSERT INTO Includes(PID, MID, Dosage, Duration) VALUES (8001, 1004, '1 tab OD', '3 da... Error Code: 1644. No valid Stock UnitPrice found for this medication at the hospital location.

Figure 6: Error message returned by the trigger

6.1.2 Insert Trigger

```
DELIMITER ||


CREATE TRIGGER Tinsert
AFTER INSERT ON Includes
FOR EACH ROW
BEGIN
    DECLARE v_HID INT;
    DECLARE v_CAID INT;
    DECLARE v_newTotal DECIMAL(10,2);

    SELECT H.HID, CA.CAID
    INTO v_HID, v_CAID
    FROM Prescription P
    JOIN ClinicalActivity CA ON P.CAID = CA.CAID
    JOIN Department D ON CA.DEP_ID = D.DEP_ID
    JOIN Hospital H ON D.HID = H.HID
    WHERE P.PID = NEW.PID;

    SELECT COALESCE(SUM(S.UnitPrice),0)
    INTO v_newTotal
    FROM Includes I
    JOIN Stock S ON I.MID = S.MID
```

```
    WHERE I.PID = NEW.PID
      AND S.HID = v_HID
      AND S.StockTimestamp = (
          SELECT MAX(StockTimestamp)
          FROM Stock
          WHERE MID = I.MID AND HID = v_HID
      );

    UPDATE Expense
    SET Total =  v_newTotal
    WHERE CAID = v_CAID;
END||


-- INSERT TEST

SELECT * FROM Expense WHERE CAID = 1001;

INSERT INTO Stock(HID,MID,StockTimestamp,UnitPrice,Qty)
VALUES (1,1003,'2025-11-27 09:20:00',35.00,50);

INSERT INTO Includes(PID,MID,Dosage,Duration) VALUES
(8001,1003,'1 tab OD','3 days');

SELECT * FROM Expense WHERE CAID = 1001;
```

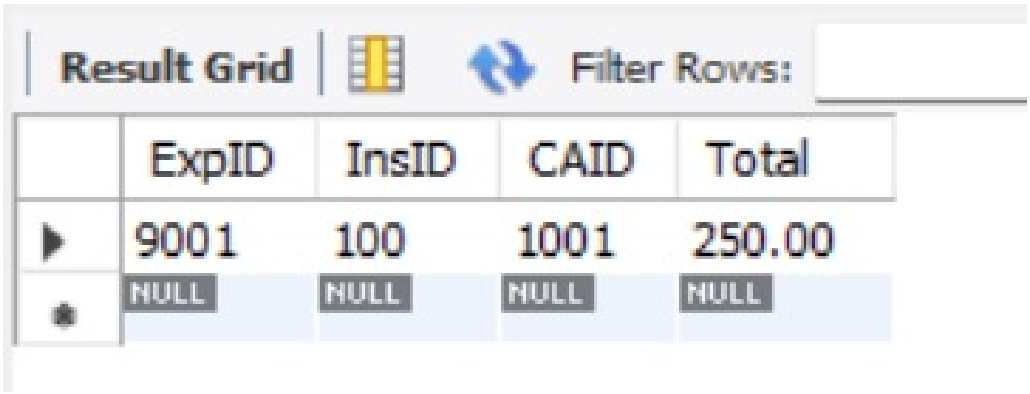

Figure 7: Initial output

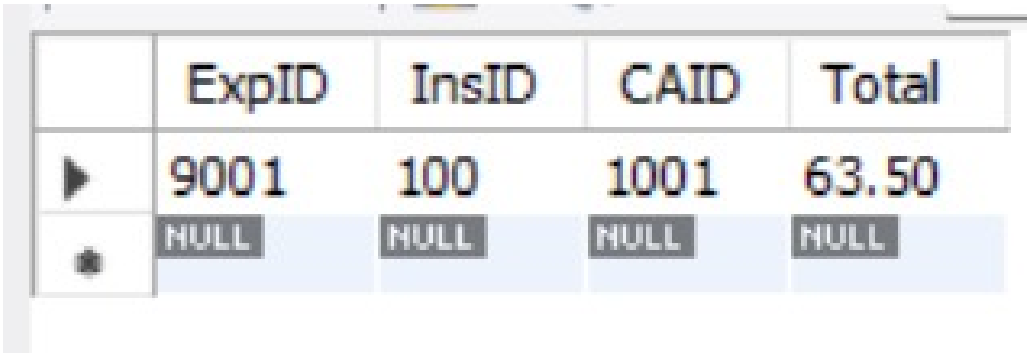

Figure 8: Error message returned by the trigger

6.1.3 TUpdate Trigger

```
CREATE TRIGGER TUpdate AFTER UPDATE ON Includes
FOR EACH ROW
BEGIN
    DECLARE v_HID INT;
    DECLARE v_CAID INT;
    DECLARE v_newTotal DECIMAL(10,2);

    SELECT H.HID, CA.CAID INTO v_HID, v_CAID
    FROM Prescription P
    JOIN ClinicalActivity CA ON P.CAID = CA.CAID
    JOIN Department D ON CA.DEP_ID = D.DEP_ID
    JOIN Hospital H ON D.HID = H.HID
    WHERE P.PID = NEW.PID
    LIMIT 1;

    SELECT COALESCE(SUM(S.UnitPrice),0)
    INTO v_newTotal
    FROM Includes I
    JOIN Stock S ON I.MID = S.MID
    WHERE I.PID = NEW.PID
      AND S.HID = v_HID
      AND S.StockTimestamp = (
          SELECT MAX(StockTimestamp)
          FROM Stock
          WHERE MID = I.MID AND HID = v_HID
      );


    UPDATE Expense SET Total = v_newTotal WHERE CAID = v_CAID;
END ||

-- UPDATE Test
SELECT * FROM Expense WHERE CAID = 1001;

UPDATE Includes
SET MID = 1004
WHERE PID = 8001 AND MID = 1001;

SELECT * FROM Expense WHERE CAID = 1001;
```

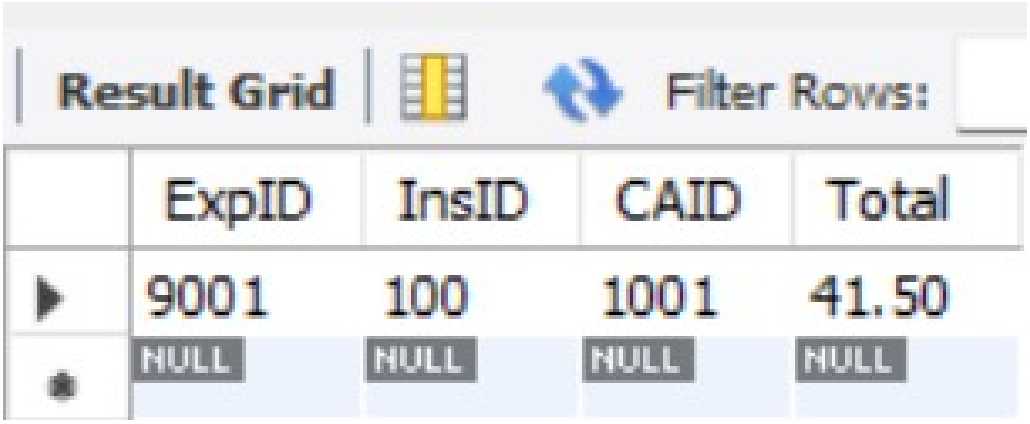

Figure 9: Error message returned by the trigger

6.1.4 TDelete Trigger

```
CREATE TRIGGER TDelete AFTER DELETE ON Includes
FOR EACH ROW
BEGIN
    DECLARE v_HID INT;
    DECLARE v_CAID INT;
    DECLARE v_newTotal DECIMAL(10,2);

    SELECT H.HID, CA.CAID INTO v_HID, v_CAID
    FROM Prescription P
    JOIN ClinicalActivity CA ON P.CAID = CA.CAID
    JOIN Department D ON CA.DEP_ID = D.DEP_ID
    JOIN Hospital H ON D.HID = H.HID
    WHERE P.PID = OLD.PID
    LIMIT 1;

    SELECT COALESCE(SUM(S.UnitPrice), 0) INTO v_newTotal
    FROM Includes I
    JOIN Stock S ON I.MID = S.MID AND S.HID = v_HID
    WHERE I.PID = OLD.PID
    AND S.HID = v_HID
    AND S.StockTimestamp = (
          SELECT MAX(StockTimestamp)
          FROM Stock
          WHERE MID = I.MID AND HID = v_HID
      );

    UPDATE Expense SET Total = v_newTotal WHERE CAID = v_CAID;
END ||

-- DELETE Test

DELETE FROM Includes
WHERE PID = 8001 AND MID = 1003;

SELECT * FROM Expense WHERE CAID = 1001;
```

	ExpID	InsID	CAID	Total
▶	9001	100	1001	6.50
*	NULL	NULL	NULL	NULL

Figure 10: Error message returned by the trigger

7 Trigger 3: Prevent negative or inconsistent stock.

7.1 Trigger Definition

```
DELIMITER ||
CREATE TRIGGER Stock_Insert_Check
BEFORE INSERT ON Stock
FOR EACH ROW

BEGIN

    IF NEW.Qty < 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Insert rejected: Qty cannot be negative';
    END IF;

    IF NEW.UnitPrice <= 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Insert rejected: UnitPrice must be greater
           than 0';
    END IF;

    IF NEW.ReorderLevel < 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Insert rejected: ReorderLevel cannot be
           negative';
    END IF;
END ||

DELIMITER ||

CREATE TRIGGER Stock_Update_Check
BEFORE UPDATE ON Stock
FOR EACH ROW
BEGIN
    IF NEW.Qty < 0 THEN
            SIGNAL SQLSTATE '45000'
            SET MESSAGE_TEXT = 'Update rejected : Stock quantity cannot
               be negative';
    END IF;
    IF NEW.UnitPrice <= 0 THEN
        SIGNAL SQLSTATE '45000'
```

```
        SET MESSAGE_TEXT = 'Update␣rejected:␣UnitPrice␣must␣be␣greater␣
            than␣0';
    END IF;
    IF NEW.ReorderLevel < 0 THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Update␣rejected:␣ReorderLevel␣must␣be␣>=␣0'
            ;
    END IF;
END ||

-- QUERY EXAMPLE FOR TRIGGER 3

INSERT INTO Stock (HID, MID, StockTimestamp, UnitPrice, Qty,
   ReorderLevel)
VALUES (3, 1002, NOW(), 7.00, -10, 15);

INSERT INTO Stock (HID, MID, StockTimestamp, UnitPrice, Qty,
   ReorderLevel)
VALUES (4, 1003, NOW(), 0, 50, 10);

UPDATE Stock SET Qty = -5 WHERE HID = 1 AND MID = 1001;
```

7.2 Trigger Output

Figure 11: Error message returned by the trigger

8 Trigger 4: Protect referential integrity on patient delete.

8.1 Trigger Definition

```
DELIMITER ||
CREATE TRIGGER PatientDeleteIntegrity
BEFORE DELETE ON Patient
FOR EACH ROW
BEGIN
   IF (SELECT COUNT(*)
      FROM ClinicalActivity as c
      WHERE c.IID = OLD.IID) > 0 THEN
      SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Cannot␣delete␣Patient␣:␣dependent␣activities␣
          still␣exists␣';
  END IF;
END  ||
```

```sql
-- QUERY EXAMPLE FOR TRIGGER 4
DELETE FROM Patient
WHERE IID = 1;
```

8.2 Trigger Output

Figure 12: Error message returned by the trigger