Views

RECALLS:

View

Dynamic result of one or more relational operations operating on the base relations to produce another relation.

- Virtual relation that does not actually exist in the database but is produced upon request, at time of request.

Contents of a view are defined as a query on one or more base relations.

- Any operations on view are automatically translated into operations on relations from which it is derived.

CREATE VIEW view_name [(column_name [...])] AS subselect ...

Example 1 - Create Horizontal View

CREATE VIEW manager3_staff
AS SELECT *
FROM staff
WHERE bno = 'B3';

SG17 Ani 81 George St, Glasgow PA1 2RI 0141-398-3545 Sex: F 10-Nov-60 1200.00 WLA12345AC B3
SG14 David Ford 65 Ardley St, Partick, Glasgow G11 0141-398-2117 Dept: M 24-Mar-58 18000.00 WTE23095O B3
SG3 Sean Bond 5 Ot Western Rd, Glasgow G12 0141-398-2040 Manager: F 3-Jun-40 24000.00 WKS00892E B3

(3 rows)

Example 2 - Create Vertical View

CREATE VIEW staff3
AS SELECT sno, fname, lname, address, tel_no, position, sex
FROM staff
WHERE bno = 'B3';

Or:

CREATE VIEW staff3
AS SELECT sno, fname, lname, address, tel_no, position, sex
FROM manager3_staff;

Advanced SQL - 1

Advanced SQL - 2

Advanced SQL - 3

Advanced SQL - 4
Example 2 - Create Vertical View

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>surname</th>
<th>address</th>
<th>tel_no</th>
<th>position</th>
<th>sex</th>
</tr>
</thead>
<tbody>
<tr>
<td>SG10</td>
<td>Ann</td>
<td>Beech</td>
<td>81 George St, Glasgow PA1 2AR</td>
<td>0141-848-3345</td>
<td>Sr. Asst</td>
<td>F</td>
</tr>
<tr>
<td>SG14</td>
<td>David</td>
<td>Ford</td>
<td>63 Asby St, Patrick, Glasgow G12</td>
<td>0141-339-2177</td>
<td>Deputy</td>
<td>M</td>
</tr>
<tr>
<td>SG5</td>
<td>Susan</td>
<td>Brand</td>
<td>5 Gl. Western Rd, Glasgow G12</td>
<td>0141-334-2001</td>
<td>Manager</td>
<td>F</td>
</tr>
</tbody>
</table>

SQL - DROP VIEW

DROP VIEW view_name [RESTRICT | CASCADE]

- Causes definition of view to be deleted from the database.
- For example: DROP VIEW manager3_staff;
- With CASCADE, all related dependent objects are deleted; i.e. any views defined on view being dropped.
- With RESTRICT (default), if any other objects depend for their existence on continued existence of view being dropped, command is rejected.

Restrictions on Views

- SQL-92 imposes several restrictions on creation and use of views.
  - (a) If column in view is based on an aggregate function:
    - Column may appear only in SELECT and ORDER BY clauses of queries that access view.
    - Column may not be used in WHERE nor be an argument to an aggregate function in any query based on view.
Restrictions on Views

(b) Grouped view may never be joined with a base table or a view.

- For example, Staff_Prop_Cnt view is a grouped view, so any attempt to join this view with another table or view fails.

View Updatability

- Consider view Staff_Prop_Cnt:

  CREATE VIEW staff_prop_cnt (branch_no, staff_no, cnt)
  AS SELECT s.bno, s.sno, COUNT(*)
  FROM staff s, property_for_rent p
  WHERE s.sno = p.sno
  GROUP BY s.bno, s.sno;

- If change definition of view and replace count with actual property numbers:

  CREATE VIEW staff_prop_list (branch_no, staff_no, property_no)
  AS SELECT s.bno, s.sno, p.pno
  FROM staff s, property_for_rent p
  WHERE s.sno = p.sno;

View Updateability

- SQL-92 specifies the views that must be updatable in system that conforms to standard.

  - Definition given is that a view is updatable iff:
    - DISTINCT is not specified.
    - Every element in SELECT list of defining query is a column name and no column appears more than once.
    - FROM clause specifies only one table, excluding any views based on a join, union, intersection or difference.
View Updatability

- WHERE clause does not include any nested SELECTs that reference the table in FROM clause.
- There is no GROUP BY or HAVING clause in the defining query.
  - Also, every row added through view must not violate integrity constraints of base table.
  - For view to be updatable, DBMS must be able to trace any row or column back to its row or column in the source table.

Integrity Enhancement Feature (IEF)

We consider 5 types of integrity constraints:
- Required data
- Domain constraints
- Entity integrity
- Referential integrity
- Enterprise constraints

1. Required Data
   position VARCHAR(10) NOT NULL

2. Domain Constraints
   (a) CHECK
      sex CHAR NOT NULL
      CHECK (sex IN ('M', 'F'))
   (b) Using DOMAIN
      CREATE DOMAIN sex_type AS CHAR
      CHECK (VALUE IN ('M', 'F'));
      sex sex_type NOT NULL

IEF - Entity Integrity

3. Entity Integrity
   - Primary key of a table must contain a unique, non-null value for each row.
   - PRIMARY KEY clause can be specified only once per table. Can still ensure uniqueness for alternate keys using UNIQUE.

4. Referential Integrity
   - If foreign key contains a value, that value must refer to existing row in parent table.
Example 3 - CREATE TABLE

CREATE TABLE property_for_rent (  
  pno PROP_NUMBER NOT NULL,  
  rooms PROPERTY_ROOMS NOT NULL DEFAULT 4,  
  rent PROPERTY_RENT NOT NULL,  
  ono OWNER_NUMBER NOT NULL,  
  bno BRANCH_NUMBER NOT NULL,  
  PRIMARY KEY (pno),  
  FOREIGN KEY (ono) REFERENCES owner ON DELETE NO ACTION ON UPDATE CASCADE);  

IEF - Referential Integrity

- CASCADE: Delete row from parent and delete matching rows in child, and so on in cascading manner.  
- SET NULL: Delete row from parent and set FK column(s) in child to NULL. Only valid if FK columns are NOT NULL.  
- SET DEFAULT: Delete row from parent and set each component of FK in child to specified default. Only valid if DEFAULT specified for FK columns  
- NO ACTION: Reject delete from parent. Default.

IEF - Referential Integrity

FOREIGN KEY (sno) REFERENCES staff  
ON DELETE SET NULL  
FOREIGN KEY (ono) REFERENCES owner  
ON UPDATE CASCADE

IEF - Referential Integrity

- Any INSERT/UPDATE that attempts to create FK value in child table without matching candidate key value in parent is rejected.  
- Action taken that attempts to update/delete a candidate key value in parent table with matching rows in child is dependent on referential action specified using ON UPDATE/ and ON DELETE subclauses:  
  - CASCADE - SET NULL,  
  - SET DEFAULT - NO ACTION.
IEF - Enterprise Constraints

Enterprise Constraints

Example:
CREATE TABLE property_for_rent(
    pno ..., address ..., 
sno VARCHAR(5)
    CONSTRAINT staff_not_handling_too_much
    CHECK (NOT EXISTS (SELECT sno
             FROM property_for_rent
             GROUP BY sno
             HAVING COUNT(*) > 10)),
    bno ...
);

ALTER TABLE

- Add a new column to a table.
- Drop a column from a table.
- Add a new table constraint.
- Drop a table constraint.
- Set a default for a column.
- Drop a default for a column.

Example 4 - ALTER TABLE

ALTER TABLE staff
    ALTER position DROP DEFAULT;
ALTER TABLE staff
    ALTER sex SET DEFAULT 'F';
ALTER TABLE property_for_rent
    DROP CONSTRAINT staff_not_handling_too_much;
ALTER TABLE renter
    ADD pref_area VARCHAR(15);

Access Control - Privileges

- Usually, every user has an associated password.
- Actions user permitted to carry out on given base table or view:
  SELECT Retrieve data from a table.
  INSERT Insert new rows into a table.
  UPDATE Modify rows of data in a table.
  REFERENCES Reference columns of named table in integrity constraints.
  USAGE Use domains, collations, character sets, and translations.
Example 5 - GRANT All Privileges

Give Manager full privileges to Staff table.

\[
\text{GRANT ALL PRIVILEGES} \\
\text{ON staff} \\
\text{TO manager WITH GRANT OPTION;}
\]

Example 6 - GRANT Specific Privileges

Give Admin SELECT and UPDATE on column Salary of Staff.

\[
\text{GRANT SELECT, UPDATE (salary)} \\
\text{ON staff} \\
\text{TO admin;}
\]

Example 7 - GRANT Specific Privileges to Multiple Users

Give users Personnel and Deputy SELECT on Staff table.

\[
\text{GRANT SELECT} \\
\text{ON staff} \\
\text{TO personnel, deputy;}
\]

Example 8 - GRANT Specific Privileges to PUBLIC

Give all users SELECT on Branch table.

\[
\text{GRANT SELECT} \\
\text{ON branch} \\
\text{TO PUBLIC;}
\]
Example 9 - REVOKE Specific Privileges from PUBLIC

Revoke privilege SELECT on Branch table from all users.

REVOKE SELECT
ON branch
FROM PUBLIC;

Example 10 - REVOKE Specific Privileges from Named User

Revoke all privileges given to Deputy on Staff table.

REVOKE ALL PRIVILEGES
ON staff
FROM deputy;