**Views**

- A view is just a relation, but we store a definition, rather than a set of tuples.
  
  ```sql
  CREATE VIEW YoungActiveStudents (name, grade) AS 
  SELECT S.name, E.grade 
  FROM Students S, Enrolled E 
  WHERE S.sid = E.sid AND S.age < 21
  ```

- Views can be dropped using the `DROP VIEW` command.
  
  - How to handle `DROP TABLE` if there's a view on the table?
  - `DROP TABLE` command has options to let the user specify this.

**Uses for Views**

- Views can be used to present necessary information (or a summary), while hiding details in underlying relation(s) (security).
- Views also useful for maintaining logical data independence when the conceptual schema changes.
- Can be used to precompute (materialize) results or partial results of common queries.

**Views vs. Relations**

- **Logical distinctions:**
  - *Updates not always possible* to a view
- **Physical distinctions:**
  - Relations must be physically stored somewhere
  - Views are either:
    - Computed *on-demand* (not indexable)
    - Stored physically (materialized) to enhance performance, and the DBA (or system) must manage the replication.

**Processing Queries with Views**

- For logically stored views, use query modification:
  
  ```sql
  SELECT Y * 
  FROM YoungActiveStudents Y 
  WHERE Y.name = 'Sam'
  ```

  - Could be rewritten (modified):
    ```sql
    SELECT Y * 
    FROM ( <view definition goes here> ) AS Y 
    WHERE Y.name = 'Sam'
    ```
  - or:
    ```sql
    SELECT S.name, E.grade 
    FROM Students S, Enrolled E 
    WHERE S.sid = E.sid AND S.age < 21 AND S.name = 'Sam'
    ```

**Processing Queries with Views (cont.)**

- For materialized views, updates to view data to maintain integrity can be:
  - **Immediate:** triggered by each base relation update
  - **Deferred:** allows temporary inconsistency in DB:
    - Lazy: Refresh views only as needed by queries
    - Faster updates
    - Slower queries
    - Periodic: Refresh views every n seconds (called snapshots)
    - Forced: Refresh views after every n updates to base tables

- Query is processed as if view were a physically stored relation.
Updates to Views

- Whether view is materialized or not, we can't always update a view because there may not be a unique update to base tables that reflects the update to the view.
- Single-table views are usually updateable.
- Multi-table views are more difficult. We will consider views defined using union, intersect, minus, and join.
- Assumption: WITH CHECK OPTION in force

Updates to Single-Table Views

- Selection-based views: INSERT, DELETE are mapped directly to the base relation.
- Projection-based views: view must include all fields of base relation that disallow null; base table insertion is padded with nulls.
- Aggregate views: not updateable.

```
CREATE VIEW YearAvg AS
SELECT S.year, AVG (S.gpa)
FROM Students S
GROUP BY S.year
```

Updates to Multi-Table Views

- A UNION B:
  - Inserted tuple goes into A if it satisfies A's definition and into B if it satisfies B's definition (A, B can be views or base tables; at least one must be satisfied)
  - Deleted tuples deleted from both A and B
  - Update = atomic (delete, insert) sequence
  - Example: RichEmps UNION SeattleEmps = RichSeattleEmps

- A INTERSECT B:
  - Inserted tuple goes into both A and B, assuming it satisfies definitions of both
  - Deleted tuple deleted from both A and B

- A MINUS B:
  - Inserted tuple goes into A, assuming it satisfies definition of A and doesn't satisfy B
  - Deleted tuple is deleted from A

Updates to Multi-Table Views

- A JOIN B:
  - Inserted tuple: A-portion inserted into A and B-portion inserted into B (if possible)
  - Deleted tuple: A-portion deleted from A and B-portion deleted from B
  - NOTE: May “generate” new tuples for view this way!!!

Reality …

- Those rules are as liberal as possible. Most actual systems require the following of a view definition for it to be updateable:
  - No GROUP, DISTINCT, UNION, MINUS, INTERSECT, or arithmetic
  - Update must be resolvable to specific rows in exactly one of the base tables involved in the view.
  - For deletion, only single-table views are updateable.
Summary

- Views useful for security, logical data independence, performance
- Stored logically (query modification required) or physically (materialized)
- View updates must be unambiguously mappable to base relation updates in order to be allowed.
- Most systems don’t allow as many view updates as they could

State of the Art (views)

- Views are becoming important for processing “decision support” queries
- Automated view creation and management (based on evolving workload)
- View and trigger interactions (semantics, optimization)
- Views for answering aggregation queries (query modification algorithms, etc.)
- Views to integrate multiple data sources
- Algorithms for deferred view maintenance