

DATABASE VIEWS

CHAPTER 5 (6/E)

CHAPTER 8 (5/E)

CHAPTER 5 OUTLINE

- Database views provide convenient usage
 - Virtual view realized when query uses that view
- Materialized views allow efficient re-use
 - Must be recalculated or updated when base tables change

VIEWS FOR CUSTOMIZATION

- Consider database(s) describing university's activities
 - Academic institution
 - Students, professors, classes
 - Grades, transcripts
 - Admissions, convocations
 - Alumni
 - Corporate institution
 - Finances, human resources
 - Board of Governors
 - Capital assets
 - Charitable institution
 - Donors, fundraising activities
 - Research institution
 - Granting agencies, industrial/non-profits/academic partners
 - Grants and contracts, intellectual property, licensing
- Each user group provided appropriate “subset” of the data
 - e.g., some financial/scheduling info relevant to most groups;
other info confidential
 - **View = external schema = *logical* subset of conceptual schema**
 - Underlying data *shared, not silo'd*.
- Updates must be seen by all affected users.

VIEWS (VIRTUAL TABLES)

- Consider again the query

```
SELECT title, year, genre  
FROM Film
```

```
WHERE director = 'Steven Spielberg' AND year > 1990;
```

- Returns all matching films currently in the database
- If re-run after updates, will give revised table of matches
- A **view** is an *unexecuted query* that can be run on demand.
 - Single table derived from other table(s)
 - A virtual table

USING VIEWS IN SQL

- **CREATE VIEW** command

- View name and a query to specify the contents of the view

```
CREATE VIEW Big_Earners AS
  SELECT E.Ssn AS Ssn, E.Lname AS Name,
         E.Salary AS Salary, M.Lname AS Manager
  FROM EMPLOYEE E, EMPLOYEE M
  WHERE E.Super_ssn = M.Ssn and E.Salary > M.Salary;
```

- Queries can use view as if it were a base table.

```
SELECT *
FROM Big_Earners
WHERE Salary < 100000;
```

- View always up-to-date

- (Re-)evaluated whenever a query uses the view
- Keeping it up-to-date is responsibility of the DBMS and not the user

- **DROP VIEW** command

- Dispose of a view

UPDATING A VIEW

- What if an update is applied to a view as if it were a base table?

```
CREATE VIEW Big_Earners AS
  SELECT E.Ssn AS Ssn, E.Lname AS Name,
         E.Salary AS Salary, M.Lname AS Manager
  FROM EMPLOYEE E, EMPLOYEE M
  WHERE E.Super_ssn = M.Ssn and E.Salary > M.Salary;
```

```
UPDATE Big_Earners
SET Salary = 100000
WHERE Name = 'Smith';
```

- Change corresponding tuple(s) in base table(s)
- Tuple might disappear from view!
 - **WITH CHECK OPTION** clause at end of view definition ensures new and updated tuples match view definition (else error)
- Deleting tuple from view might require update to base table instead of deletion from base table
 - e.g., deletion from CS338 view $\stackrel{?}{=}$ deletion from UW database?
- Not all views are updateable.
 - What if `Salary` defined as sum of two base attributes or as aggregate such as `SUM` or `AVG`?
 - What if `Big_Earners` defined as a `UNION` of two tables?

MATERIALIZED VIEWS

- If the base tables don't change, neither does the view instance.
 - Re-executing view definition each time view is used is wasteful if base data has not been updated.
- Solution: **view materialization**
 - Create a temporary view table when the view is first queried
 - Keep view table on the assumption that more queries using the view will follow
 - Use *materialized* view (if it exists) to answer future query
 - Requires efficient strategy for *automatically updating view table* when the base tables are updated
 - Options when any base table is updated:
 1. Delete the materialized view
 2. Rematerialize the view
 3. Incrementally update the view
 - DBMS determines what new tuples must be inserted, deleted, or modified in materialized view

SUMMARY

- Views are virtual or derived tables.
- Can be used for any query wherever base table can appear
- May or may not be updatable
 - Unions, joins, and (aggregate) functions are problematic
- Materialized views used to save query time
 - Must be kept up-to-date if base table(s) updated