CS 1555

Lecture 13

**SQL Views**

Create view

- View is a table derived from base tables and other views

- CREATE VIEW <name> AS SELECT…

- Views can be queries as if they were base tables

- It is not a table – it does not physically exist

- A view is a “virtual table” derived from base tables

- A view is a “named query”

Advantages of views

- Logical independence

- Convenience and clarity when writing queries (can be used just like tables)

- Security – different data access privileges can be given to different users

Modify & drop a view

- Modify a view: CREATE OR REPLACE VIEW CS\_STUDENT AS

- New query must generate the same schema

- Dropping a view: DROP VIEW…;

View updateability

- Updates to a view could be complex

- Need to map onto updates with defining table

- Try to update an aggregation like: AVG() 🡪 could have infinite possible values so it won’t work

- In general, a view is called updateable if all updates on the view can be unambiguously translated back to tuples in the base tables

- A view update is unambiguous if only one update on the base tables can accomplish the desired update effect

- A view is not updateable if an update on a view can be mapped to more than one possible update on the base tables

SQL standard for view updateability

- View with a single defining table is updateable if the view attributes contain the primary key

- Views defined using aggregate functions are not updateable

- Views defined on multiple tables using joins are generally not updateable

Migrating tuples

- Migrating tuples out of updateable views: an update or insert may eliminate a tuple from view

- Prevent migration with WITH CHECK OPTION – attributes in WHERE clause are read only

Efficient view implementation

- DBMS implements views in two ways: Query rewriting/modification, view materialization

- With expected tradeoffs

Query rewriting

- Presents view query in terms of a query on underlying base tables

- Disadvantage: re-compute view with each query, which is inefficient for views defined via complex queries

Materialized views

- Physically store view and its data

- Advantage: avoid re-computing view with each query

- Assumption: More queries can use the same view

- Disadvantage: materialized view maintenance is needed

- Materialized view should be updated when any base table used in view definition is updated

Updating materialized views

- Efficient strategies for automatically updating

- Avoid recomputing view from “scratch”

- Incremental update

Views vs temporary tables

- No standard, but temporary tables are:

- Visible to current SQL session

- Automatically dropped when session ends

- Cannot have FK constraints

- Temporary tables are local