Views

Author : Akash Kurup

Categories : DataBase Management Systems

Views

1. We have assumed up to now that the relations we are given are the actual relations stored in the database.
2. For security and convenience reasons, we may wish to create a personalized collection of relations for a user.
3. We use the term **view** to refer to any relation, not part of the conceptual model, that is made visible to the user as a ``virtual relation''.
4. As relations may be modified by deletions, insertions and updates, it is generally not possible to store views. (Why?) Views must then be recomputed for each query referring to them.

View Definition

1. A view is defined using the **create view** command:
   create view v **as** <query expression>

   where is any legal query expression.

   The view created is given the name
   v

   .
2. To create a view *all-customer* of all branches and their customers:
   create view *all-customer* **as**
   $$\Pi_{bname,cname}(deposit) \cup \Pi_{bname,cname}(borrow)$$
3. Having defined a view, we can now use it to refer to the virtual relation it creates. View names can appear anywhere a relation name can.
4. We can now find all customers of the SFU branch by writing
   $$\Pi_{cname}(\sigma_{bname="SFU"}(all-customer))$$

Updates Through Views and Null Values

1. Updates, insertions and deletions using views can cause problems. The modifications on a view must be transformed to modifications of the actual relations in the conceptual model of the database.
2. An example will illustrate: consider a clerk who needs to see all information in the *borrow* relation except *amount*.Let the view *loan-info* be given to the clerk:

   create view *loan-info* **as**
   $\Pi_{bname,loan\#,cname}(borrow)$

3. Since SQL allows a view name to appear anywhere a relation name may appear, the clerk can write:
   $loan\text{-}info \leftarrow loan\text{-}info \cup \{(\text{“}SFU\text{”}, 3, \text{“}Ruth\text{”})\}$

   This insertion is represented by an insertion into the actual relation *borrow*, from which the view is constructed.

   However, we have no value for *amount*. A suitable response would be

   - Reject the insertion and inform the user.
   - Insert (``*SFU*'',3,``*Ruth*'',null) into the relation.

   The symbol **null** represents a null or place-holder value. It says the value is unknown or does not exist.
4. Another problem with modification through views: consider the view
   create view *branch-city* **as**
   $\Pi_{bname,ccity}(borrow \bowtie customer)$

   This view lists the cities in which the borrowers of each branch live.

   Now consider the insertion

   $branch\text{-}city \leftarrow branch\text{-}city \cup \{(\text{“}Brighton\text{”}, \text{“}Woodside\text{”})\}$

Using nulls is the only possible way to do this (see Figure 3.22 in the textbook).

If we do this insertion with nulls, now consider the expression the view actually corresponds to:

$$\Pi_{bname,ccity}(borrow \bowtie customer)$$

As comparisons involving nulls are always **false**, this query misses the inserted tuple.

To understand why, think about the tuples that got inserted into *borrow* and *customer*. Then think about how the view is recomputed for the above query.