Module 7 Assignment

Recommend cursor processing options (SENSITIVE/INSENSITIVE, SCROLL/NO SCROLL, WITH HOLD/WITHOUT HOLD, FOR UPDATE) for the following database applications:

Online application for searching movie reviews

I recommend the database cursor have the following options:

SENSITIVE, SCROLL, WITHOUT HOLD, READ ONLY

Reason for SENSITIVE option: Changes made outside the cursor should be immediately effective to the cursor. So that users can view any live update for the movie reviews when they are searching the database. In this way, users do not need to refresh their browser every time when a new comment is added. We can leverage this sensitive cursor to provide any update for the reviews to the user.

Reason for SCROLL option: The SCROLL option allows the application to move through the review database in a non-sequential manner. When there are a thousand or million rows of reviews, we may need pagination on the UI for users to jump to different pages. In this case, the SCROLL option is useful to display results on a different page.

Reason for WITHOUT HOLD option: Using the WITHOUT HOLD option specifies that the cursor cannot be used outside of the transaction that created it. In this application, since most of the database transactions are to scan and read from the movie review database, and in most situations, they are independent of each other, so the browser can recreate and reinitialize another cursor after the previous searching transaction is finished.

Reason for READ ONLY option: READ ONLY option specifies that rows retrieved using the cursor cannot be updated. In a searching application, almost all of the queries are supposed to only access and return data rows according to the user’s requirement. So the cursor should not be updatable.

Online shopping application (searching products, adding them to an electronic shopping card, and then checking out).

I recommend the cursor to have the following options:

SENSITIVE, SCROLL, WITHOUT HOLD, FOR UPDATE

Reason for SENSITIVE option: With the SENSITIVE option, changes made outside the cursor should be immediately effective to the cursor. An example to illustrate the importance of the SENSITIVE option would be: When there is someone else who just bought the product and make the remaining inventory decrease to 0, other users who are putting that product in their shopping cart should be able to know that this product is not available anymore. So the SENSITIVE option is useful here.

Reason for SCROLL option: If there are a lot of products held in the shopping application, pagination or filtering functions are required for users to quickly access their desired product rather than requiring the user to view each product result row by row. So SCROLL option is useful here as it provides the ability to move through the database in a non-sequential manner.

Reason for WITHOUT HOLD option: Under the WITHOUT HOLD option, after a transaction is committed, the cursor should be automatically closed. In an online shopping application, there are some important functions which include payment using a credit card or other payment providers like Paypal. According to my understanding, payment should be an atomic operation, i.e. either complete success or failure. If we set the option to be WITH HOLD, then there may be some problems if we are accessing a failed transaction and incorrectly proceeding with it, or processing another transaction while the previous transaction is not completed.

Reason for FOR UPDATE option: Under FOR UPDATE OF <column list> option, rows retrieved using the cursor can be updated or deleted. In an online shopping application, there may be some functions that need to update or delete some data rows, for example, after the user completes the purchase of a product, the application needs to decrease the inventory number of the product. So FOR UPDATE can be used here.

Web-based application for the maintenance of a company’s product database

SENSITIVE, SCROLL, WITH HOLD, FOR UPDATE

Reason for SENSITIVE option: With the SENSITIVE option, changes made outside the cursor should be immediately effective to the cursor. In a web-based application for product database maintenance, if there are other transactions committed outside the cursor, the web UI should be able to reflect this change. Otherwise, users may need to refresh the browser to initialize a new cursor for viewing this update.

Reason for SCROLL option: The SCROLL option provides the ability to move through the database in a non-sequential manner. In the aforementioned application, as it needs to provide the function of pagination and filtering for users to view and update the product data efficiently, so SCROLL option is useful here.

Reason for WITH HOLD option: Under WITH HOLD option, after a database transaction is committed, the cursor should remain open. In this application, the possible database transaction would be to update rows of product data according to different conditions. After committing the transaction, the cursor can remain open so that no need to redefine and reinitialize the cursor again, saving time for continuous processing of the later transaction.

Reason for FOR UPDATE option: The application involves updating or deleting rows of data to perform the maintenance of the product database, so FOR UPDATE option is needed here.

Application to maintain your personal music and playlist library

SENSITIVE, SCROLL, WITH HOLD, FOR UPDATE

Reason for using the SENSITIVE option: If there are some transactions committed outside

the cursor, it should be visible to the cursor. So that the application

can show live updates if any music/playlist is inserted/updated/modified.

Reason for using the SCROLL option: There may be some functional requirements from the

users to access the data in a non-sequential manner. For example, the user may want

to play a random song in his/her playlist or retrieve the playlist on a different

page. Therefore, using the SCROLL option for the cursor is suitable.

Reason for using WITH HOLD option: WITH HOLD option allows the cursor to stay open after the transaction is committed. If a song or playlist is inserted/deleted, the cursor can remain open to perform the remaining transaction, so no need to reinitialize or redefine the cursor again.

Reason for using FOR UPDATE option: To enable the application to maintain the music

and playlist library, the cursor needs to have edit rights on the data.

Therefore, FOR UPDATE option is suitable in this example.

Describe how you would handle transactions and locking for an online banking application.

In an online banking application, the transaction need to be implemented correctly, as to prevent the following data phenomena: dirty read, nonrepeatable reads and phantom reads. We can set the isolation level to SERIALIZABLE to prevent all of these data phenomena. Although there may bring possible performance penalty, this is necessary to protect the data integrity in an online banking application.

I would use Explicit mode to handle the transaction, as it provides a higher flexibility for developers to control where to rollback and commit the transaction.

In terms of handling locks, we need to prevent deadlocks. We can do deadlock detection by timing lock waits and giving up after a preset time interval. If the transaction runs after the preset time interval, the request needs to be terminated, and roll back to allow other request to proceed.

Also, we can add an application lock to the application, that is to handle the lock by application logic rather than the DBMS. For example, we can assign the timestamp of the last update to the row of data.

State whether you would recommend XML or JSON for the following data structures, along with your reasons:

* + Employment history that can be transmitted to potential employers
    - * JSON would be suitable in this case, as it supports array and it is relatively easy to read.
  + Storage of application metadata that describes how reports and charts are laid out on a web page
    - * XML would be suitable in this case, as it is widely used for representing data structures on web service.
  + The list of music contained in your personal music library
    - * JSON would be suitable in this case as it can support array.