2023-12-15 Lab Assignments

SQL Part 4

Views, Functions, Triggers, Temporal Data, Bitmap

1. SportsClub - View

- Create a view that displays for each targetGroup the number of courses that is assigned to the targetGroup and the percentage in relation to the total number of courses.

  Example for explanation: if the total number of courses is 10 and there are 5 courses for targetGroup 'fam' and 5 courses for targetGroup 'all', then 'fam' and 'all' account for 50% of courses each.

- View is to display targetGroup, number of courses and percentage

- Is the view updateable?

knirsch@kiu.edu.ge

2. SportsClub - View

1. Create a view that displays all trainers and information about them.
   The output table is to display
   1. trainer name,
   2. trainer DoB,
   3. trainer age,
   4. trainer gender,
   5. trainer email,
   6. trainer entryDate,
   7. trainer licence
   8. and trainer startTeach datee.
2. Is the view updatable?
3. If so, update the view: In the view, change the email of trainer lazy and add a startTeach date for him. Does it work? How do you have to do this?

knirsch@kiu.edu.ge

3. University Database - Trigger

Write a trigger on the table university.examination which inserts a row into a log table whenever a row in the table examination is updated. (Exams and grades are final. No one should update / change a row in the examination table. So, a trigger documents if there are changes.)

- Create a table log_exam with the necessary columns (timestamp, current user, all old values, all new values)
- Write your trigger
  (You can write the trigger using the "create Trigger interface" or you can write it directly as SQL query.)
- Test your trigger
  The result should give you the information of who changed what and when.
  The user will always be root, of course, as we are working with a single user system. You get the current user with the function user(). In the trigger you have to declare a variable and read the current user into that variable, from there insert it into your log table.

knirsch@kiu.edu.ge

4. University Database – Bitmap Index

university course table:

1. Which column would fit well for a bitmap index?
2. Write / draw the bitmap index for the column.
   Assume that the values in the existing table are the only possible values.

Task is taken from last year's final exam.

knirsch@kiu.edu.ge

5. View on bank table

Have another look at our bank table. The first digit of the bank number identifies the clearing area a bank belongs to. (Clearing is needed if money transfers do not go through. In case that numbers are wrong or mistyped or twisted the transfer order is analyzed by a clearing center.).

1. Create a view that displays the clearing area (1st digit) and the number of banks that belong to each clearing area.

2. Query the view and display the minimum number of banks and the maximum number of banks that belong to a clearing area.

knirsch@kiu.edu.ge

6. Sytem Versioned Table

System-versioned tables that have a lot of updates might get very large because all the historic data adds up. This can have a negative effect on performance.

Does mariadb allow for separation of the current data from historic data that are stored together in one table? (Attention: here we are looking at separation on a physical NOT on a logical level!) If this would be possible it would reduce the size and make queries especially on current data faster.

Consult mariadb documentation for a solution. Try the solution on a versioned table.

1. Create a system versioned table for the membership fees.
2. Populate your table with some versions of data.
3. Verify that mariadb actually implements the separartion between historic and current data. For verification you need to query the database information_schema, table paritions. Write the query.

knirsch@kiu.edu.ge

7. University Database – User Defined Function

1. Add a column studentEmail to the student table.
2. Write a user-defined function in the university database that generates the following email for a name given as input parameter:
   name@kiu.edu.ge
   example: Jonas@kiu.edu.ge
   Recommendation: User HeidiSQL functionality ("Create new stored function) to do this.
3. Improve your user-defined function in such a way that it can handle duplicate names. The first Jonas gets the mail address Jonas@kiu.edu.ge, the second Jonas gets the mail address Jonas1@kiu.edu.ge, the third Jonas gets the mail address Jonas2@kiu.edu.ge, and so on. (Limit can be 10 as we only want to test the principle.)
4. Write a trigger on the table student that fires each tine a new student is inserted. The trigger calls the function and sets the value for the column studentEmail.
5. Test function and trigger.

Additional links

- https://mariadb.com/kb/en/programming-customizing-mariadb/
- https://github.com/nomemory/hr-schema-mysql