Homework 4

Group 14: Sara Rydell & Filip Northman

Tasks (P)

Task 1

We didn't receive feedback that we had to change anything from our previous homework.
We also don't want to make any additional changes.

Our previous homework schemas:

- Book(ISBN:integer, Title:string, Edition:string, Language:string, Publisher:string, Publication_Date:date, Prequel_ISBN:integer)
- Book_Genre(ISBN:integer, Genre:string)
- Author(Author_ID:integer, Author_Name:string)
- Book_Writer(ISBN:integer, Author_ID:integer)
- Physical_Book(Physical ID:integer, ISBN:integer, Damage:integer)
- User(User ID:integer, Full_Name:string, Email:string, Address:string)
- Student(Student_ID:integer, Programme:string)
- Admin(Admin_ID:integer, Department:string, Phone_Number:integer)
- Loan(Borrowing_ID:integer, Physical_ID:integer, User_ID:integer, Borrowing_Date:date, Max_Time:integer, Return_Date:date)
- Fine(Borrowing_ID:integer, Amount:integer)
- Transaction(Transaction_ID:integer, Borrowing_ID:integer, Amount:integer, Payment_Date:date, Payment_Method:string)

Task 2

a) **Tuple-based CHECK constraints** make constraints on a row, which means that multiple columns in each tuple get checked and could be something like date_of_return can't be before date_of_borrowing since that would be impossible.

b) **Attribute-based CHECK constraints** make constraints on a column, which means that individual attributes get checked, such as that date of birth often can't be a string or be before today's date.

c) **Triggers** are used to specify when to check a constraint and exactly what to do and have three parts: when an event happens (ex. update to an attribute) the system will check a condition (constraint or query) and if satisfied an act will be performed (ex. deletion/update/insertion).

Task 3

1. Tuple-based check constraints check an entire row (tuple) whereas attribute-based check constraints only check a column (attribute).

2. Tuple based CHECK constraints are less time efficient than attribute-based CHECK constraints when querying as in the case of tuples, comparisons have to be made between values from multiple columns instead of just checking the input value. This could be valuable in a big database to save time so it's not good to use tuple-based constraints unnecessarily.

Task 4

One **tuple-based CHECK constraint** in our database could be that it shouldn't be possible for books to have the return_date be before the borrowing_date as you cannot return a book before you have even borrowed it.

One **attribute-based CHECK constraint** is to check so that all added email addresses have an @kth.se part in them as that is required for it to be valid, but that may be a mute point as the mail addresses are given by the school.

Task 5

a) Difference between views and regular queries:
   Virtual relations (tables) are views. Regular queries such as relations are stored, although virtual views are not stored in the database but can be queried as if they existed. A view can be seen as a tailored picture of a database and it can be composed of many tables, can be modified, removed and materialized.

   It's more favorable to use views in cases when:
   You need to get an overview of the queries made, you can then use a view to see how the tables interact for a specific group of users.
   Like for a table, you can grant permission to users through a view that contains specific data that the users are authorized to see. Thus views also provides a consistent layer protecting the data of underlying tables from unwanted changes.

b) Difference between a virtual view and a materialized view:
   Views are not capable of storing any data, they only point to the data that is viewed whereas a materialized view are capable of storing data which makes accessibility to data a faster process. A view is virtual or logical memory that is based on the 'select'

query whereas a materialized view provides access to duplicate, physical data in a separate table.
Therefore a materialized view is favorable to use when it is something that will be shown multiple times and therefore is better to save so the process is quicker and safer.
A virtual view is better to use when you only want to view less often or maybe want to update the data often and it is useless to save.

Task 6

a) **Accepted**. Check constraints will always accept if the check constraint returns true or null and only be rejected if it returns false. Thus it will pass when compared to an integer since this comparison returns null.
b) **Accepted**. Two null values are not considered equal when existing in the same table and that means even in the presence of a unique constraint it is possible to store duplicate rows that contain a null value in at least one of the constrained columns.
c) **Accepted**. With a tuple-based check constraint, comparing a null value with a non null value it will be accepted as the check constraint will always pass if it returns true or null and only fail if it returns false. Thus it will pass when compared to other data which is not null since it returns null.
d) **Rejected.** Primary keys can be viewed as a combination of not null and unique constraints so it wouldn't be passed in this case since null can't be accepted.
e) **Rejected.** This is because a foreign key in a relation containing null values cannot match the values of a parent key, since a parent key by definition can have no null values.

Tasks (P+)

Task 1

One benefit is that the functions are simpler and thus less likely to cause problems in the future and gives us a faster database. Since more advanced constraints are not often required this restriction helps the programmer with implementing restrictions without messing up their database.

One disadvantage is that more advanced check statements are hard to implement with these restrictions. Therefore for more complex restrictions multiple check statements are often needed. One example is that you can forcefully implement check statements which check several rows or even tables, the problem here though is that if you change or input a value into the other row or table which violates this check it will drop the entire table. Which would be a catastrophe for a company.

Task 2

In our case our database could in a good way make use of views since we will have to create different access levels for the different kinds of users (being students and admins). Here creating different views would solve the requirements about access and editing rights mentioned in the study case and keep the database secure. It would also be good to create standardized materialized views for viewing book information since these functionalities will be excessively used and also borrowing of the books since only selected information should be added to the database by the users. One negative is that using materialized views will take more resources, although this problem shouldn't be enough to not use them, but enough to be careful.