SFWRENG 3DB3 – FALL 2021

Assignment 1

Submitted by: Hamrish Saravanakumar (400246228)

1. Data independence, when described as a property of DBMS, aids the ability to change the schema of a database at one level of the database system without having to change the schema at a higher level as well. It has two properties, the logical schema, where information is stored about how the data is managed inside, and the physical schema, where the data is stored in bit format in order to change the physical data without impacting the logical data. This system is well-represented in practice through the concepts of storage systems. For example, SSD has become a more efficient replacement for hard-disks. When a replacement of hard-disks with SSD takes place, data independence ensures that it will not have any impact on the logical data.

2. BRANCH (branchName, address, city, sales, manager)

- Primary Key: Address
    - It is assumed that since no two establishments/buildings can have the same address, no two branches could possibly share addresses. It is assumed in this case that branchName is not a unique identifier for each branch, but rather a common name that could potentially be repeated

CUSTOMER (cID, firstName, lastName, address, city, birthDate, phoneNum,totalAssets)

- Primary Key: cID
    - Assuming that the customer ID is unique for every single customer, the ID on its own would be a sufficient identifier to differentiate between every customer, thus making it a suitable primary key

LOAN (loanNum, branchName, amount, duration, interest)

- Primary Key: loanNum
    - A loan number, according to the Canadian mortgage association, is a unique identifier for ay given loan. This loan does not depend on the branch that it was opened at, so this would mean that loanNum would be a suitable unique identifier for a loan, and would thus be a suitable primary key

BORROWER (cID, loanNum)

- Primary Key: cID, loanNum
    - A borrower is only a borrower such that there is a loanNum attached for the individual to borrow from, and that the individual is a registered customer with an ID. This means that neither of these two fields can be null, and it is important to uniquely identify a borrower entry that both of the attributes are considered primary keys. This is because a customer can have multiple loans AND a loan can be taken out by multiple customers (joint-loans).

ACCOUNT(acctNum, branch, balance, type)

- Primary Key: acctNum
    - According to various financial institutions, the account numbers specific to a bank are unique and do not recycle over time, and these account numbers, although have a branch number attached, are not dependent on the branch to be unique. Therefore, the account number on its own (acctNum) would be a suitable primary key that uniquely identifies entries in the ACCOUNT table

DEPOSITER(cID, acctNum)

- Primary Key: cID, acctNum
    - A depositer is only a depositer such that there is a account attributed for the individual to deposit to, and that the individual is a registered customer with an ID. This means that neither of these two fields can be null, and it is important to uniquely identify a depositer entry that both of the attributes are considered primary keys. This is because a customer can have multiple accounts (credit, debit, etc.) AND an account can belong to multiple customers (joint-account, student account, etc.)

Referential Integrity Constraints:

- loanNum of the BORROWER relation is a foreign key for loanNum of the LOAN relation
- acctNum of the DEPOSITER relation is a foreign key for acctNum of the ACCOUNT relation
- cID of the BORROWER relation is a foreign key for cID of the CUSTOMER relation
- cID of the DEPOSITER relation is a foreign key for cID of the CUSTOMER relation

For example, by implementing this foreign key, we can ensure that a borrower can only exist such that a loan exists. If a loan is completely paid off or is cancelled in some way, meaning that the loanNum is either removed or updated, then any entry in the BORROWER relation must be updated or removed as well. The same logic applies for each of the other foreign keys that would be set up. The “on delete cascade” functionality would fit best for this database, as a borrower or depositor with a particular cID or loan/acctNum should not exist if either of these attributes are null