COMP3278A Introduction to Database Management Systems

2021 – 2022

Due date: 23:59pm Oct. 25, 2021

Assignment 1: ER Design and SQL

Question 1 (50%) [Entity-Relation Design]

A HKU-DBMS team is building an online shopping system. Please read the below users' requirements, and answer questions (a) and (b).

User Requirements.

- The main purpose of the system is to manage transactions of cash between accounts.
- Each user in the system is identified by a unique username. Each user has a profile attribute, which consists of a birthday, an email address, and the date the user joined the system.
- Users have two specifications, i.e. seller and customer. A user can be a seller and a customer at the same time.
- A customer can form many groups and this customer would have the role of leader. A customer can join several groups as member.
- A seller sells one or many products. A product has a unique product ID and product name.
- A product contains several items. Each item has an item ID and can be uniquely identified when combing the item ID and corresponding product ID. An item has the attributes of item name and price.
- A user (including seller and customer) owns one or more accounts. Each account in the system is identified by a unique account ID, and an account must be owned by a user. The user can login the account via account name and password.
- There are two specifications of accounts, including cash account and point account. An account is either a cash account or a point account.
- Cash account is an account that keeps cash in the system. The remaining cash is denoted as balance.
- Point account has no cash by itself. Instead, it has point in the account. The point can be earned by taking parting in some activities. It can be transformed to the cash and stored in cash account. Point account must be linked to only one cash account. A cash account may or may not have multiple point accounts.
- A transaction is created by one account with the attribute of create date. It must be related to only one item type. One transaction can buy the same item many times, thus it has the attribute of quantity and the amount price can be derived.
- Each transaction is uniquely identified by a transaction ID. It is possible to leave (attach) several messages along with the transaction.

- In fact, each transaction is completed by using one or more internal transactions. An internal transaction has an internal ID, which can only uniquely identify each internal transaction when combining with the corresponding transaction ID.
- Each internal transaction is used to specify the cash flow from one cash account to another cash account.

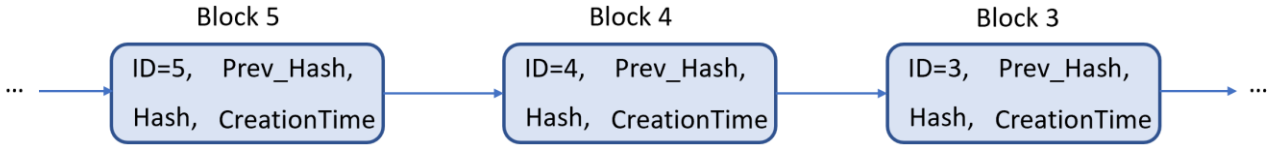
(a) Please model the above users' requirements by drawing the corresponding E-R Diagram.
    - If a requirement is unclear, a reasonable assumption could be made. You must **clearly state** all such assumptions.

(b) Please reduce your E-R Diagram into relational table schemas. For each relation, underline the primary key and identify all foreign key(s), if any.

Question 2 (50%) [SQL Query]

Please read the below background story and table schemas, and answer the query tasks by writing SQL.

Background Information. A simple blockchain is a linked list of blocks as shown in the below figure, where each block contains a unique block ID, a hash (a unique string of characters), a hash of the previous block, and a creation time of the block. Each block is used to store transaction data, which are records of transferring digital currency. A unit of the digital currency is called a token. Each token can be uniquely identified by a token ID and has a market value.



Blockchain Management Affiliation (BMA) wants to build a transaction system, aiming to manage the transactions of tokens between accounts. You are hired to join them in preparing the database for the system.

The tables of the database are shown and explained as follows. *(Note: the primary key for each table is indicated in orange and the foreign keys are indicated by red arrows.)*

- Each account in the system is identified by a unique account number, and an account must be owned by a user.
- A token must be owned by only one account. Each token can be uniquely identified by token ID and has its market value.
- A transaction is initiated by specifying a total amount to be transferred from one account to another account. Each transaction is uniquely identified by a transaction ID.
- In fact, each transaction is completed by using one or more internal transactions. An internal transaction has an internal ID, which can only uniquely identify each internal transaction when combining with the corresponding transaction ID.

- Each internal transaction is used to transfer one token from one token account to another token account.
- A transaction is kept in only one block. Each block has a unique block ID, and a hash value is calculated and recorded in each block.
- A block is always linked to only one previous block (except for the very first block in the system), in order to form a blockchain. The hash value of the previous block is also explicitly recorded in a block. It is possible that multiple blocks have the same previous block at the same time.

Block

block_id	hash	prev_hash	prev_id
B3	34fs	a67d	B2
B4	5g8y	34fs	B3
B5	f4h6	34fs	B3

InternalTransaction

trans_id	internal_id	from_account	to_account	token_id
1	1	A1	A2	T1
1	2	A1	A2	T2
2	1	A2	A1	T3

Token

token_id	value	account_num
T1	120	A1
T2	200	A1
T3	80	A2
T4	150	A3
T5	50	A3

Account

account_num	user_name
A1	Alice
A2	Alice
A3	Bob

Transaction

trans_id	amount	from_account	to_account
1	320	A1	A2
2	90	A2	A1

Figure: tables with data types

Account (account_num, user_name)

Token (token_id, value, account_num)
Foreign keys: account_num refers **Account**

Transaction (trans_id, amount, from_account, to_account)
Foreign keys: from_account refers **Account**, to_account refers **Account**

InternaTransaction (trans_id, internal_id, from_account, to_account, token_id)
Foreign keys: trans_id refers **Transaction**, from_account refers **Account**, to_account refers **Account**, token_id refers **Token**

Block (block_id, hash, prev_hash, prev_id)
Foreign keys: prev_id refers **Block**

Please write one SQL query for each of the following questions from (a) to (e).
- Your query must handle all possible scenario according to the given constraint.
- There should not be duplicated results.
- A1_test_tables.sql and A1_test_data.sql on Moodle are sample data for your reference

(a) Find the hash value of the block(s) that has more than one next block.
(b) Find the transaction ID of the transaction(s) where the total token value does not match with the transaction amount.
(c) Find internal transaction(s) that is produced by 'Alice' and the transferred token value is smaller than 100. Return internal ID and its token value.
(d) Find the token ID of the token(s) that has never been transferred.
(e) Show the username and the number of accounts the user owns for the user(s) that has the greatest number of tokens with value greater than 100.

Submission Requirement. Please submit your answers in a single PDF file on Moodle before the deadline. Late penalty will be applied for late submission. Plagiarism tools will be applied to each submission.
For each query task in Question 2, please submit your SQL query and a screenshot (screen capture) of executing the SQL query with returned results in MySQL.

[End of Paper]