**Homework Assignment 3**

Submission due: 11:59PM June 11, 2020   
*(We will solve the problems together in class on June 9)*

|  |
| --- |
| * This is an individual work; Please be clear with HGU CSEE Standard:   + Submitting assignments or program codes written by others or acquired from the internet without explicit approval of the professor is regarded as cheating.   + Showing or lending one’s own homework to other student is also considered cheating that disturbs fair evaluation and hinders the academic achievement of the other student.   + It is regarded as cheating if two or more students conduct their homework together and submit it individually when the homework is not a group assignment. * Read the assignment carefully. *In this assignment, you will need to* ***write and execute several SQL queries****; and* ***submit the results of your queries***. * You are **allowed to re-use any of the queries from the lecture slides** while developing solutions to the problems. * When finished, submit your work to *LMS.* |

**1. Read, understand Slides 18-24, and answer the following questions.**

(a) What is denormalization?

A strategy used on a previously-normalized database to unnormalize database tables to increase performance

- Denormalization is a process of trying to improve the read performance of a database, at the expense of losing some write performance, by adding redundant copies of data

(b) Describe the difference between functions and stored procedures.

Functions have a return value, but Procedures don't have returns

(c) When is it NOT advised to use triggers?

Risk of unintended execution of triggers, for example, when

- Loading data from a backup copy

- Replicating updates at a remote site

- Trigger execution can be disabled before such actions

Other risks with triggers:

- Error leading to failure of critical transactions that set off the trigger

- Cascading execution

(d) What is page?

Fixed-size(varies by platform) data block

The unit of transfer between disk and memory

Page can contain tuples, meta-data, indexes, log records, ...

(e) What are the pros and cons of log-structured organization for database storage?

Log-structured organization

- Pros

Faster writes, easy rollbacks

- Cons

Reads can take longer time

(f) What is the difference between the FLOAT/DOUBLE and NUMERIC/DECIMAL data types?

FLOAT/DOUBLE - Variable Precision

NUMERIC/DECIMAL - Fixed Precision

(g) What is system catalog?

Meta-data stored inside of a DBMS as a table format(schema meta-data)

(h) Compare the *N*-ary storage model and decomposition storage model.

NSM : stores all attributes for a single record continuously in a page

- Fast insert, updates, and deletes

- Good for queries that needs the entire tuple

- Not good for scanning a small subset of the attributes of a very large table

DSM : stores the values of a single attribute for all records contiguously in a page

- Reduces the amount wasted I/O

- Better query processing and data compression

- Slow for point queries, inserts, updates, and deletes(as all records are split over pages)

(i) In buffer management, what are the dirty flags, reference counter, and pin?

Dirty flag: single bit indicating if the page has been modified since it has been read from disk

Reference counter: the number of threads or queries in memory that want the page

Pin: pages that have not been safely written back to disk yet.

(j) Compare the page directory and page table.

The page directory is the mapping from page IDs to page locations in the database files

- All changes must be recorded on disk to allow the DBMS to find on restart

The page table is the mapping from page ids to a copy of the page in buffer pool frames

- This is an in-memory data structure that does not need to be stored on disk

(k) Why does sequential flooding occur? and give a potential solution discussed in class.

Sequential flooding

- cause : The case where each page request causes an I/O; caused by LRU + repeated sequential scans

- Sol : LRU-K, Buffer localization, Priority hints

**2. Match each of the following key types to the corresponding definition.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Super key | ● |  | ● | Unique identifier attribute(s) chosen in a relation |
| Candidate key | ● |  | ● | Minimal subset of uniquely identifiable attribute(s) |
| Alternate key | ● |  | ● | Attribute(s) that define(s) the relationship between relations |
| Composite key | ● |  | ● | Any uniquely identifiable attribute(s) |
| Foreign key | ● |  | ● | Any key with more than one attribute |
| Primary key | ● |  | ● | Minimal subset of uniquely identifiable attribute(s) that are NOT chosen to represent rows in a relation |

**3. Describe the operation of the following trigger:**

|  |
| --- |
| **CREATE TRIGGER** *credits\_earned* **AFTER UPDATE OF** *takes* **ON** (*grade*) **REFERENCING NEW ROW AS** *nrow* **REFERENCING OLD ROW AS** *orow* **FOR EACH ROW WHEN** *nrow.grade* <> 'F' **AND** *nrow.grade* **IS NOT NULL  AND** (*orow.grade* = 'F' **OR** *orow.grade* **IS NULL**) **BEGIN ATOMIC  UPDATE** *student* **SET** *tot\_cred* = *tot\_cred* +   (**SELECT** *credits* **FROM** *course* **WHERE** *course*.*course\_id* = *nrow.course\_id*)  **WHERE** *student.id* = *nrow.id*; **END**; |

[E] Whenever an update of the *takes* table is made,

[C] it checks for all new records with a valid grade (but previously had no grade or the 'F' grade) and automatically

[A] update the corresponding students total earned credits(tot\_cred) by adding up the newly completed(added) course credit.

**4. Consider the following hash function.**

|  |  |  |  |
| --- | --- | --- | --- |
|  | *key* | *Value* | *h*(*key*) |
| *h*: *key* → *h*(*key*) | A | 308 | 0 |
| B | 201 | 2 |
| C | 74 | 2 |
| D | 39 | 3 |
| E | 3 | 1 |
| F | 127 | 1 |

(a) Now consider that we have a hash table of capacity 7. Assuming keys are inserted in the order of A 🡪 B 🡪 C 🡪 D 🡪 E 🡪 F and the *linear probing* hashing scheme is used, draw the final state of the hash table below (note that each entry of the hash table stores both a key and an associated value).

[0] A|308

[1] E|3

[2] B|201

[3] C|74

[4] D|39

[5] F|127

[6]

(b) Again, considering that we have a hash table of capacity 7 and assuming that keys are inserted in the order of A 🡪 B 🡪 C 🡪 D 🡪 E 🡪 F, if the *Robin Hood* hashing scheme is used, how does the operation of the hash table change? Draw the final state of the hash table below.

[0] A|308[0]

[1] E|3[0]

[2] F|127[1]

[3] C|74[1]

[4] B|201[2]

[5] D|39[2]

[6]