8th May, 2021

CS-GY 6083 - B, Spring-2021.
Principles of Database Systems.

FINAL Exam

FINAL EXAM [100 points with 40% weight]
TIME LIMIT: 2 hours and 30minutes:
05/08/2021 10:00 AM to 12:30 PM EST
Please read instructions carefully before writing exam

- **Write your name, student id, and net id below**
- **Last Name:** **First Name:**
- **Net ID:** **Student ID:**

THIS IS AN ONLINE EXAM. PLEASE LOGIN TO ZOOM MEETING USING YOUR NET ID (MANDATORY, DO NOT LOGIN WITH YOUR PERSONAL EMAIL ACCOUNT). THE ZOOM MEETING ID: 980 4846 4508.Passcode: 501852
https://nyu.zoom.us/j/98048464508?pwd=dk95S1lZb1diU2RiM2k3dkJvZXRUQT09
(This is our regular Saturday sessions Zoom meeting)

- **WRITE YOUR ANSWERS UNDER EACH QUESTION IN THIS WORD DOCUMENT AND SUBMIT IT ON OR BEFORE 12:15PM TO NYU CLASSES > ASSIGNMENTS > FINAL_EXAM. Save and submit the exam submission document in format <Your Net id>_Final_Spring_2021_6083B. YOU MAY RESUBMIT YOUR ASSIGMENT ONCE BEFORE THE SUBMISSION DEADLINE OF 12:15PM. PLEASE MUTE YOUR MICROPHONE DURING ENTIRE EXAM DURATION.**

- **This exam has 5 sections A, B.C.D, E and each section has multiple questions. All sections and questions have grading points. There is NO negative points for any wrong answers. All sections and questions are expected to answer.**

- **IF YOU HAVE ANY QUESTION DURING THE EXAM, PLESAE SEND YOUR QUESTION PRIVATELY TO PROFESSOR ON ZOOM MEETING CHAT WINDOW. DO NOT SPEAK IN MICROPHONE.**

- **USE Oracle Data Modeler for ERD diagram, no hand drawing. Insert snapshot of database design models in same Word/PDF document. NO ZIP FILE WILL BE ACCEPTED. NO ANY TYPE OF COPY WILL BE GRADED.**

GOOD LUCK!

A) Answer following questions briefly [30 points]

i) **Which disk organization techniques (RAID level) is suitable for the database files and archived log files and why? Explain in context of characteristics of RAID levels of your answer.**
Database Files: Level 6 will be preferred because it has low update rate. It offers P+Q redundancy. Which is even higher than Level 5.

Log Files: Level 10 is preferred. Because this is preferred for high update rate application. Also, this has mirroring as well as striping. Log files are frequently updated so Level 10 is preferred.

Different schemes of disk organization are used to achieve performance and reliability:
RAID level 0: It is used for high performance, no mirroring only block is stripped.
RAID level 1: Blocks are mirrored to achieve redundancy
RAID level 1 +0: Blocks are both mirrored and stripped
RAID level 2: Additional parity bit is stored for error detection and correction. Memory style error correction code with bit stripping.
RAID level 3: When data is written a parity, bit is calculated and stored in parity bit disk as a single parity bit is enough of error correction.
RAID level 4: Block Interleaved parity, a separate parity block is kept on separate disk for corresponding blocks from rest of the disks. On every write, block parity bits are calculated and written to parity disk.
RAID level 5: Block interleaved distribute parity, partitions data and parity are stored among all disks instead of storing parity on a specific parity disk.
RAID level 6: Same as raid level 5, but here instead of storing one parity block, and additional mirrored parity block information is redundantly stored to guard against multiple disk failure.

[Students are not required to describe all RAIL levels.]

ii) What is database dead lock? Give an example of dead lock situation. Explain at least three strategies for dead lock prevention.

Deadlock is a situation when two or more transactions have put a lock on a common shared resource and each one of them is waiting for others to release their locks first. For example, Transaction A might hold a lock on some rows in the *Accounts* table and needs to update some rows in the *Orders* table to finish. Transaction B holds locks on those very rows in the *Orders* table but needs to update the rows in the *Accounts* table held by Transaction A. Transaction A cannot complete its transaction because of the lock on *Orders*. Transaction B cannot complete its transaction because of the lock on *Accounts*. All activity comes to a halt and remains at a standstill forever unless the DBMS detects the deadlock and aborts one of the transactions.

The three possible methods to prevent a dead lock are:

- Wait- die scheme – Rollback the younger transaction, older transaction may wait for the younger transaction to release the data item. Here older transaction means a transaction with smaller time stamp.
- Wound- wait scheme – Older transaction forces rollback of younger transaction, younger transaction may wait for older ones.
- Timeout-Based scheme – Break the deadlock on detection. Wait for the lock for a specified time and if lock is not granted then rollback the transaction and redo.

[Students may have another method, i.e. Locking all records that are intended to update in advance and thus putting exclusive lock on all such records and releasing them after updates are done.
e.g. SELECT FOR UPADATE statement before actual UPDATE
SELECT * FROM EMP FOR UPDATE WHERE DEPTNO=10;
UDATE EMP SET SAL=SAL+10 WHERE DEPTNO=10;
COMMIT;
VS.
UPDATE EMP SET SAL=SAL+10;
COMMIT;]

iii) Explain the difference between WHERE clause and HAVING clause of SQL statement with an example.

Both WHERE and HAVING clause filters the result sets, however, WHERE clause is used to apply filter conditions on row data stored in the table, and HAVING clause is used to apply filter conditions on aggregate data (from the result of GROUP BY clause). HAVING clause can be applied only if there is a GROUP BY clause, and WHERE clause can be applied regardless of GROUP BY clause.

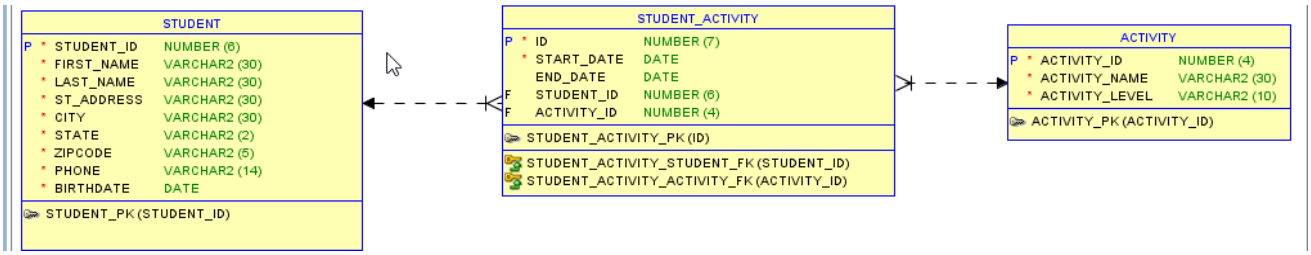
Example:

```
SELECT empno, count(*)
FROM ap_emp
WHERE sal > 3000
GROUP BY empno;

SELECT deptno, job, AVG(sal)
FROM ap_emp
GROUP BY deptno, job
HAVING AVG(sal)>3000;
```

B) SQL queries [30 points]

Consider following relational model. Create tables in Oracle/MySQL, populate tables with attached DML file "STUDENT_ATIVITY_DML.SQL", and write SQL queries to answer questions. Tables need to be created with your initial as prefix, e.g. AP_STUDENT. You will need to replace table names with your table names in the DML files.



Submit SQL queries and corresponding result screenshots.

I. List first name and last name of those students who have not participated in any activity

```
SELECT FIRST_NAME, LAST_NAME
FROM STUDENT_ACTIVITY a RIGHT OUTER JOIN STUDENT b ON
a.STUDENT_ID = b.STUDENT_ID
WHERE ACTIVITY_ID IS NULL;
```

FIRST_NAME	LAST_NAME
Katherine	Watson
Ravish	Kumar

Download CSV

2 rows selected.

II. List student id, activity name, activity level, activity duration in number of months

```
SELECT b.STUDENT_ID, c.ACTIVITY_NAME, c.ACTIVITY_LEVEL,
ROUND(MONTHS_BETWEEN(END_DATE, START_DATE))
ACTIVITY_DURATION
FROM STUDENT_ACTIVITY a JOIN STUDENT b on a.STUDENT_ID =
b.STUDENT_ID
JOIN ACTIVITY c on a.ACTIVITY_ID = c.ACTIVITY_ID;
```

STUDENT_ID	ACTIVITY_NAME	ACTIVITY_LEVEL	ACTIVITY_DURATION
100000	Piano Concert	MODERATE	0
100003	Math Contest	CHALLENGE	0
100005	Art Gallery	EASY	1
100008	Coding Competition	CHALLENGE	0
100003	Good Voice NYU	MODERATE	2
100001	City Trip	EASY	1
100000	Volunteer Trip	MODERATE	0
100001	Best Choreographer	CHALLENGE	3
100004	Italian Style Dinner	EASY	0
100003	Introduce to PS	MODERATE	0
100004	Piano Concert	MODERATE	1
100005	Piano Concert	MODERATE	0
100006	Math Contest	CHALLENGE	0
100003	Math Contest	CHALLENGE	0
100008	Art Gallery	EASY	2
100009	Art Gallery	EASY	3
100000	Coding Competition	CHALLENGE	0
100000	Good Voice NYU	MODERATE	3

100000	Good Voice NYU	MODERATE	3
100001	City Trip	EASY	0
100001	Volunteer Trip	MODERATE	7
100003	Best Choreographer	CHALLENGE	3
100003	Italian Style Dinner	EASY	0
100005	Introduce to PS	MODERATE	3
100005	Italian Style Dinner	EASY	0
100006	Best Choreographer	CHALLENGE	3
100005	Volunteer Trip	MODERATE	12
100009	City Trip	EASY	0
100009	Good Voice NYU	MODERATE	7
100008	Piano Concert	MODERATE	0
100006	Math Contest	CHALLENGE	0

Download CSV

30 rows selected.

III. List name of activity along with total number of participations for that activity which has highest number of participation

```
SELECT b.ACTIVITY_NAME, a.cnt "NUMBER OF PARTICIPATIONS" FROM
(SELECT ACTIVITY_ID, cnt, RANK() OVER (ORDER BY cnt DESC) rnk FROM
(SELECT ACTIVITY_ID, COUNT(*) cnt FROM STUDENT_ACTIVITY GROUP BY
ACTIVITY_ID)) a
JOIN ACTIVITY b ON a.ACTIVITY_ID = b.ACTIVITY_ID
WHERE a.rnk = 1;
```

ACTIVITY_NAME	NUMBER OF PARTICIPATIONS
Math Contest	4
Piano Concert	4

Download CSV

2 rows selected.

IV. List activity names and their level for those activities which have highest average duration

```
SELECT b.ACTIVITY_NAME, b.ACTIVITY_LEVEL FROM
(SELECT ACTIVITY_ID, ave_dur, RANK() OVER (ORDER BY ave_dur DESC) rnk
FROM
```

```
(SELECT ACTIVITY_ID, AVG(ACTIVITY_DURATION) ave_dur FROM (SELECT
c.ACTIVITY_ID, ROUND(MONTHS_BETWEEN(END_DATE, START_DATE))
ACTIVITY_DURATION
FROM STUDENT_ACTIVITY a JOIN STUDENT b on a.STUDENT_ID =
b.STUDENT_ID
JOIN ACTIVITY c on a.ACTIVITY_ID = c.ACTIVITY_ID) GROUP BY
ACTIVITY_ID)) a
JOIN ACTIVITY b ON a.ACTIVITY_ID = b.ACTIVITY_ID
WHERE a.rnk = 1;
```

ACTIVITY_NAME	ACTIVITY_LEVEL
Volunteer Trip	MODERATE

Download CSV

V. List student id, first name, and last name of top 3 students in terms of number of activities they have participated

```
SELECT STUDENT_ID, FIRST_NAME, LAST_NAME FROM
(SELECT a.STUDENT_ID, b.FIRST_NAME, b.LAST_NAME, COUNT(*) FROM
STUDENT_ACTIVITY a JOIN STUDENT b ON a.STUDENT_ID = b.STUDENT_ID
GROUP BY a.STUDENT_ID, b.FIRST_NAME, b.LAST_NAME ORDER BY 4 DESC)
WHERE ROWNUM <= 3;
```

STUDENT_ID	FIRST_NAME	LAST_NAME
100003	Anthony	Simmons
100005	Irene	Miller
100000	John	Garcia

Download CSV

3 rows selected.

C) Index [15 point]

Consider the relational model in question B.
Assume that, there are 30,000 students, 100 activities, 3 activity levels, and total 400,000 combination of students and activities.
The following query is frequently used by the department.

```
SELECT a.student_id, a.first_name, a.last_name,
```

```
        months_between(b.end_date. b..start_date), c.activity_name,
        c.activity_level
FROM student a JOIN student_activity b ON a. student_id=b.student_id
      JOIN activity c ON b.activity_id=c.activity_id
WHERE a.state in ('NY', 'NJ','CT') and c.activity_level='EASY' and
        upper(activity_name)=' Piano Concert';
```

Answer following questons.

i) **Which attributes are most suitable for index to improve the performance of this query?**

For Student Table: State

For Studnet_Activity Table: Student_id, and Activity_id

For Activity Table: Activity_Level and UPPER(activity_name)

ii) **What type of index is appropriate for each column that you have intended to create index and why?**

For Student Table: State (Bitmap Index is more suitable, since few distinct values)

For Studnet_Activity Table: Student_id, and Activity_id (Normal Index on each of these columns, since these columns are used for join conditions, and index is suitable for columns used in join conditions. The same columns in Student and Activity table are already having index since they are PK in those tables)

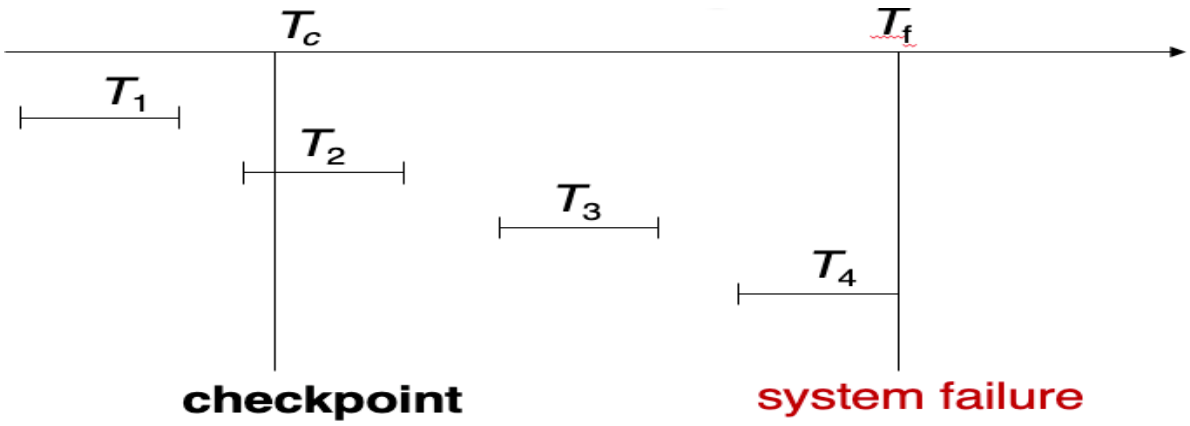
For Activity Table: Activity_Level (Bitmap Index is more suitable, since few distinct values) UPPER(activity_name) (since function is used along with column in where clause, function based index is suitable)

iii) **Write DDL command to create the index for each column that you have intended to create index.**

```
CREATE BITMAP INDEX IDX_STATE ON SK_STUDENT(STATE);
CREATE INDEX idx_studnet_activiry_a_id  ON
STUDENT_ACTIVIRY(activiry_id) ;   (note that, index name is  to be less
than 30 chars)
CREATE INDEX idx_studnet_activiry_s_id  ON
STUDENT_ACTIVIRY(student_id) ;    note that, index name is  to be less
than 30 chars)
   CREATE BITMAP INDEX idx_activity_level ON Activity(activity_level);
   CREATE  INDEX idx_activity_name ON Activity(Upper(Activity_name));
```

D) Check Point and Recovery [15 points]

Consider following scenario of transactions for a problem set



Transactions T1, T2, T3, and T4 occurred in chronological order. The checkpoint in database happened at given time Tc and later on time Tf, the system crashed on power failure.

Following is the details of work done in each transaction.

Transaction T1	Transaction T2	Transaction T3	Transaction T4
< T1 start>	< T2 start>	< T3 start>	< T4 start>
< T1, X, 5000, 3000 >	< T2, X, 3000, 2500 >	< T3, A, 1800, 1200 >	< T4, X, 3000, 4000 >
< T1, Y, 3000, 2000 >	< T2, Y, 2000, 1000 >	< T3, B, 800, 600>	< T4, Y, 2000, 2500 >
< T1, Z, 1000, NULL >	< T2, Z, NULL, 500 >	< T3, C, 250 , NULL >	< T4, Z, NULL, 700 >
<T1, COMMIT >	<T2, ROLLBACK>	<T3, COMMIT >	< T4, A , 1200, 800 >
			< T4, B, 600, 1200>

i) **Upon system recovery, which transaction(s) will undergo REDO operations and which transactions will undergo UNDO operations and why?**
T2 and T3 will undergo REDO because they are completed between checkpoint and failure. T4 will undergo UNDO because it has not finished before the failure.

ii) **For transaction(s) that will undergo UNDO, what will be written out in transaction log?**
Undo T4 :

<T4, B, 600>
<T4, A,1200>
<T4, Z, NULL>
<T4, Y, 2000>
<T4, X, 3000>

<T4, abort>

iii) What will be data values of data items X, Y, Z, A, B, and C after the system is recovered.
X : 3000, Y : 2000, Z : NULL, A : 1200, B : 600, C : NULL

E) Transactions [10 points]

Consider following database activities with 'Deferred Modification' for commits.

I. Connect to the database
II. SELECT empno, deptno, sal FROM ap_emp WHERE deptno=30;
III. UPDATE ap_emp SET sal=sal+300 WHERE deptno=30;
IV. ALTER TABLE ap_emp ADD (birthdate date);
V. ROLLBACK;
VI. SELECT empno, deptno, sal FROM ap_emp WHERE deptno=30;
VII. SELECT empno, deptno, sal FROM ap_emp WHERE deptno=20;
VIII. UPDATE ap_emp SET sal=sal+300 WHERE deptno=20;
IX. COMMIT;
X. ROLLBACK;
XI. SELECT empno, deptno, sal FROM ap_emp WHERE deptno=20;
XII. SELECT empno, deptno, sal FROM ap_emp WHERE deptno=10;
XIII. UPDATE ap_emp SET sal=sal+100 WHERE deptno=10;
XIV. Disconnect from database (EXIT)
XV. Connect to the database

a) For each new transaction give name to the transaction, e.g. TX1, TX2, TX3. etc., and state when each transaction started and when ended by filling up the following chart.

Transaction Number	Started At Activity Number	Ended At Activity Number
TX1	I	IV

TX2	IV	V
TX3	V	IX
TX4	IX	X
TX5	X	XIV
TX6	XV	
TX7		

b) **What changes will you see as applied when you connect to the database at step XV?**

UPDATE ap_emp SET sal = sal + 300 WHERE deptno = 30;
ALTER TABLE ap_emp ADD (birthdate date)
UPDATE ap_emp SET sal = sal + 300 WHERE deptno = 20;
(Students may have written in words, instead of stamments, e.g. employees in department 30 and department 20 will have salary increase of $300, and employee table will have new column birthdate added)