CSE 414 Final Exam Review
(Adapted from the autumn 2018 final)

Problem 1: Warm Up

Select either True or False for each of the following questions.

a) When turning an E/R diagram into a relational schema, every entity and relation in the diagram is represented by a table.

True False

b) In schema design, a key is a superkey with the minimal number of attributes among all superkeys for a given table.

True False

c) An unclustered index can never speed up a query more than a clustered index on the same attributes, assuming only one index on the table.

True False

d) In database concurrency, "isolation" means that no matter the actual schedule executed by the system, the effect should be the same as if the transactions run one after the other.

True False

e) Strict 2PL (assuming row updates but not insert/deletes) guarantees conflict serializable schedules but not durability to crashes.

True False

Problem 2: Transactions

a) Draw the precedence graph of the following schedule. Is this schedule conflict serializable? If so, write YES and the order in which the transactions would have to run in the equivalent serial schedule. Otherwise write NO.

$R_1(A);\ R_3(B);\ W_1(A);\ R_2(A);\ R_1(B);\ R_3(A);\ W_1(B);\ W_2(A);\ R_3(A);$

b) Read the schedule below and answer the following questions. A and B are each a field of a tuple in the database. You can assume writes to the database are only updates, not insert/deletes.

Time	Transaction 1	Transaction 2
1		Begin transaction
2	Begin transaction	
3	Lock(A)	
4		Lock(B)
5	Read(A)	
6		Read(B)
7		Unlock(B)

8	Lock(B)	
9	Unlock(A)	
10	Read(B)	
11	Write(B)	
12	Unlock(B)	
13	Commit	
14		Commit

This schedule follows the rules of two-phase locking.

True False

This schedule follows the rules of <u>strict</u> two-phase locking.

True False

This schedule is serializable.

True False

Problem 3: Schema Design

Consider this schema for a database of movies. (There are some extra attributes added since the midterm.) The primary keys are underlined.

ACTOR (pid, fname, lname, agency)
MOVIE (mid, name, year)
DIRECTOR (did, fname, lname, studio)
CASTS (pid, mid, role)
MOVIE_DIRECTORS (did, mid)

A tuple in `Casts` represents the relationship that a person from the `Actor` table with `pid` starred (was cast) in a `Movie` with `mid`. Similarly, `Movie_Directors` represents the relationship that a person from `Director` with `did` directed a `Movie` with `mid`.

a) Draw an E/R diagram for the movie database including entities, relationships, and attributes. **Include a constraint that each movie has exactly one director.**

Schema repeated here for your reference:

ACTOR (pid, fname, lname, agency)
MOVIE (mid, name, year)
DIRECTOR (did, fname, lname, studio)
CASTS (pid, mid, role)
MOVIE_DIRECTORS (did, mid)

For the next problems, assume that the `Director` relation is only the following 4 tuples:

did	fname	lname	studio
0	Sofia	Coppala	Paramount
1	Ang	Lee	MGM
2	Steve	McQueen	Warner
3	Francis	Coppala	Paramount

b) Circle whether the following functional dependencies hold on this instance of `Director`.

`fname` → `lname` Holds / Does not hold

`lname` → `studio` Holds / Does not hold

`studio` → `fname` Holds / Does not hold

c) Write two new tuples to add to the database such that none of the three functional dependencies listed above will hold on the new instance.

did	fname	lname	studio

Problem 4: BCNF Decomposition

Assume the following functional dependencies hold on the relation R:

R (A, B, C, D, E, F)

A -> CDE
F -> ABD
D -> E

a) Write the closures of the following attributes:

$\{A\}^+$ = _____

$\{B\}^+$ = _____

$\{AB\}^+$ = _____

$\{F\}^+$ = _____

b) Decompose R into Boyce-Codd Normal Form with respect to the above functional dependencies, indicating the new relations and their attributes (for example R1(A, B...), R2(B, C...)

Problem 5: Indexing and Query Optimization

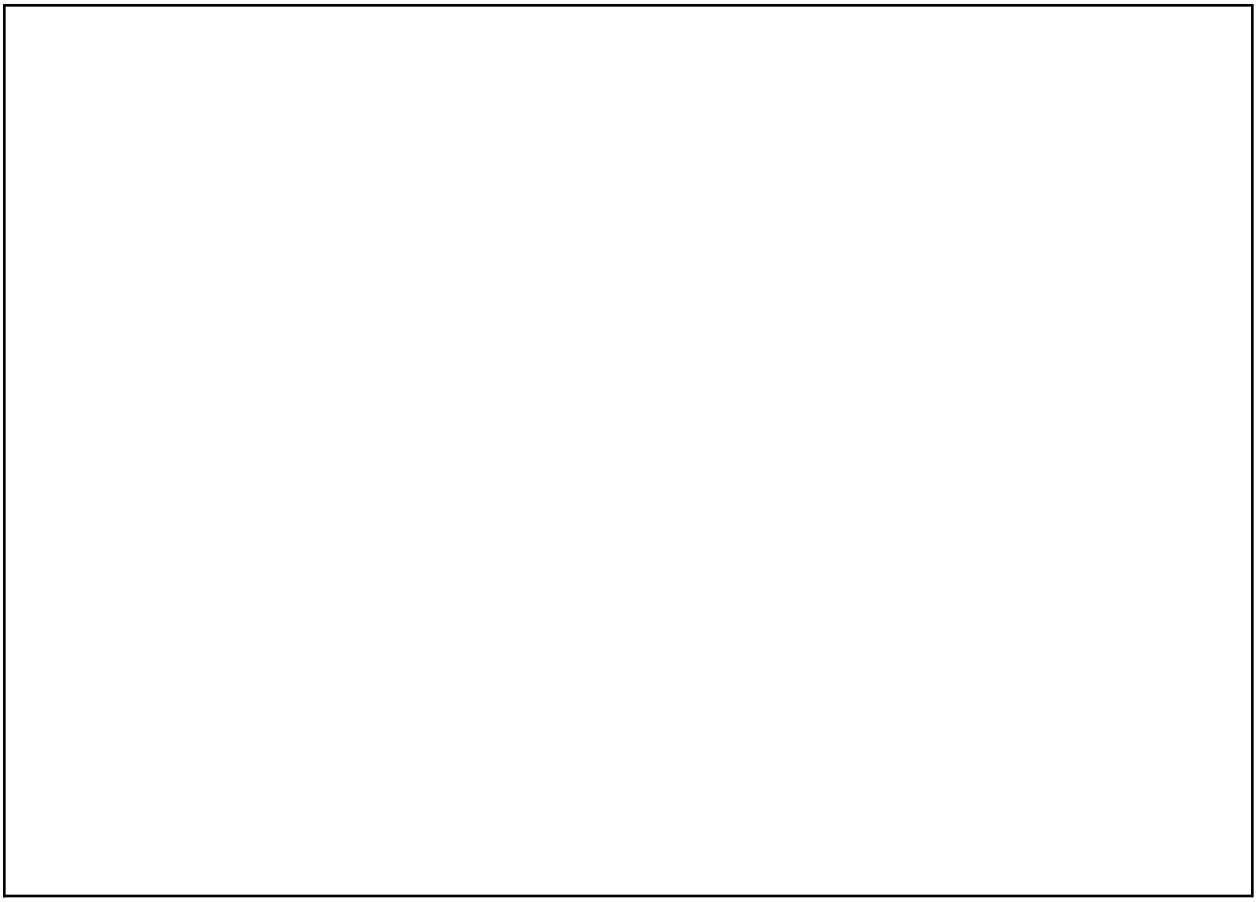
Assume we have relations R and S in a database, with the following attributes. The primary keys are underlined.

```
R(a, b)
S(c, d)
```

Say we run the following SQL query on the above tables:

```
SELECT R.a, S.c
FROM   R, S
WHERE R.a = S.d AND
      R.b = '1234'
```

Write a physical query plan for the above query that uses a block nested loop join (assume we use the page-at-a-time refinement.) Your query plan should be a relational algebra tree with annotations for the physical operators.



Recall the notation for specifying statistics on the tables in a database:
`T(X)` is the number of tuples in a relation `X`.
`B(X)` is the number of blocks the relation `X` takes up on hard disk.
`V(X, y)` is the number of distinct values for the attribute `y` in the relation `X`.

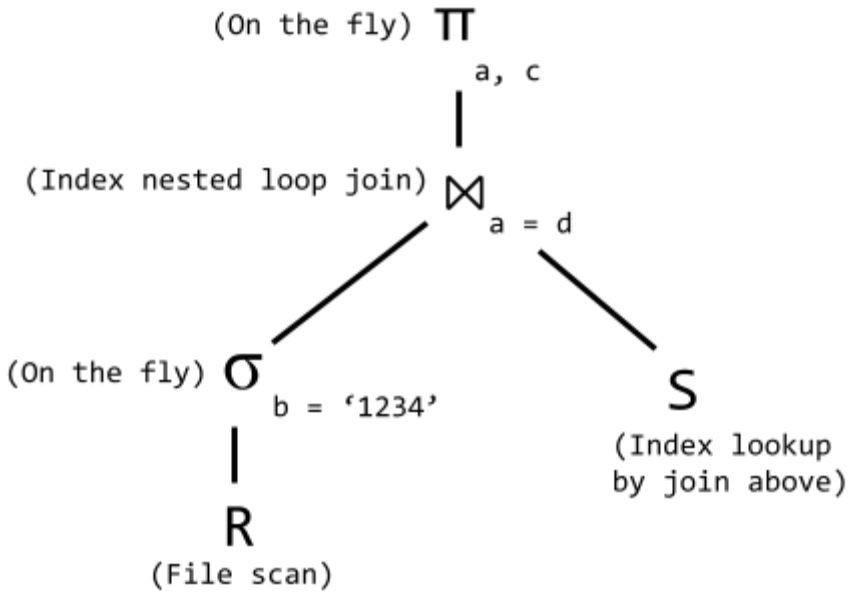
Now assume we have the following statistics on the tables `R` and `S`.

R	S
T(R) = 10,000 B(R) = 500 V(R, a) = 400 V(R, b) = 2,000	T(S) = 10,000 B(S) = 500 V(S, c) = 500 V(S, d) = 1,000 Unclustered B+ tree index on S(d)

b) Write the estimated cost in disk block I/Os to run your query plan with the block nested loop join on the previous page.

_____ I/Os

Now consider this physical plan for the same SQL query:



c) Write the estimated cost in disk block I/Os for the above query plan.

_____ I/Os

Problem 6: Parallel Databases

Say you are designing a parallel relational database to store purchase data for manufacturing products. The tables are:

```
Manufacturer(mid, name, category, city, state)
Purchase(pid, mid, date, amount) -- mid is a foreign key to Manufacturer
```

Tuples in the purchase table record individual payments in dollar amounts to a manufacturer for purchases of some product. There is a large amount of data in both tables that would have to be spread between multiple machines. As the database designer, you know that the most common query that will be run on the system is:

```
SELECT m.mid, SUM(p.amount) AS total_revenue
FROM   Purchase p, Manufacturer m
WHERE  p.mid = m.mid
GROUP BY m.mid
```

a) Describe in a few sentences how you would partition the data between machines if your goal is to maximize performance of the above query.

b) Now consider that instead of maximizing performance of any query, your goal is to minimize skew and store the data evenly across the machines. Describe in a few sentences how you would partition the data in that case.