

# CS W186 Fall 2018 Final

**Do not turn this page until instructed to start the exam.**

## Contents:

- You should receive one *double-sided answer sheet* and a 21-page *exam packet*.
- The final has *10 questions*, each with multiple parts.
- The final is worth a total of *126.5 points*.

## Taking the exam:

- You have *170 minutes* to complete the final.
- All answers should be written on the answer sheet. The exam packet will be collected but not graded.
- For each question, place only your *final answer* on the answer sheet; do not show work.
- For multiple choice questions, please *fill in the bubble or box completely* as shown on the left below. *Do not mark the box with an X or checkmark.*



- Use the blank spaces in your exam for scratch paper.

## Aids:

- You are allowed **two** 8.5" × 11" double-sided pages of notes.
- The **only** electronic devices allowed are basic scientific calculators with simple numeric readout. No graphing calculators, tablets, cellphones, smartwatches, laptops, etc. are allowed.

## 1 FDs & BCNF (12 points)

1. (5 points) Decompose  $R = ABCDEFGH$  into BCNF in the order of the given functional dependencies:  $F = \{C \rightarrow A, B \rightarrow EF, H \rightarrow BCG, F \rightarrow CD, G \rightarrow B\}$ .

Which of the following tables are included in the final decomposition?

- A. AC
- B. BCDGH
- C. BCGH
- D. BG
- E. DH

2. (3 points) Two students are writing a decomposition of tables based on the attribute set  $R = ABCDEF$  and the functional dependency set  $F = \{C \rightarrow D, A \rightarrow B, B \rightarrow EF, F \rightarrow A\}$ .

A Stanford student decides to write a decomposition of tables as ABDE and ABCF. Right now, this is not a lossless decomposition. Which of the following changes (applied individually, not combined together) would make this a lossless decomposition?

- A. Adding  $D \rightarrow ABCDEF$  to the functional dependency set F
- B. Adding  $E \rightarrow C$  to the functional dependency set F
- C. Adding  $B \rightarrow D$  to the functional dependency set F

3. (4 points) Please mark the functional dependencies that are consistent with the following table.

A	B	C	D
1	3	6	2
2	4	1	1
1	3	5	2
2	2	1	2

- A.  $A \rightarrow D$
- B.  $AB \rightarrow D$
- C.  $A \rightarrow CD$
- D.  $ABC \rightarrow D$

## 2 Transactions/2PL (17 points)

1. (3 points) Consider the following schedule:

$T_1$	$T_2$
Read(A)	
Read(B)	
	Read(A)
	Read(B)
$A = A + B$	
	$c = \text{MIN}(A, B)$
	$d = \text{MAX}(A, B)$
$B = A - B$	
$A = A - B$	
	$A = c$
	$B = d$
Write(A)	
	Write(A)
Write(B)	
	Write(B)

Which of the following describe this schedule?

- A. Serializable
  - B. View Serializable
  - C. Conflict Serializable
2. (2.5 points) For each of the following, mark either true or false.
- A. Two Phase Locking must be used, or we cannot guarantee conflict serializability.
  - B. Two Phase Locking must be used, or we cannot guarantee serializability.
  - C. Two Phase Locking ensures that we do not have cascading aborts.
  - D. Strict Two Phase Locking ensures that we do not have deadlocks.
  - E. Strict Two Phase Locking requires that transactions acquire locks only at the start of the transaction, and release all locks only at the end of the transaction.

3. (2 points) In our new database, we implement lock promotion similarly to Homework 5: if we need to block the transaction, we place the request at the front of the queue. Unlike Homework 5, however, we release the lock a transaction originally held before blocking the transaction and adding the promotion request to the queue. We wish to test our implementation to make sure that the request was indeed placed at the front, rather than the back, of the queue, and write the following test:

```

Transaction t1, t2, t3;
t1 requests __ (1) __ lock on database
t2 requests __ (2) __ lock on database
t3 requests __ (3) __ lock on database

t2 requests a promotion to __ (4) __
check that t2 is blocked

t1 releases lock on database
check that t2 has a __ (4) __ lock on database

```

Our new database does not support multigranularity locking, so transactions can only have an S or X lock. What lock types do we need for this test to properly test that our implementation indeed places the request at the front of the queue (in other words, the test should pass if and only if we placed the promotion request at the front of the queue)? If there are multiple answers, choose any one.

(1)	(2)	(3)	(4)
A. S	A. S	A. S	A. S
B. X	B. X	B. X	B. X

4. (1.5 points) In this question, assume that the levels of granularity we have are: Database, Table, Page, Record. For each of the following, mark either true or false.
- A. To write to a page, we must have an X lock the page, the table, and the database.
  - B. A transaction that has only one lock—an IX lock on the database—has enough to request an X lock on a page.
  - C. There is no need to check for a SIX and IX conflict at the table level, because the conflict can be resolved at a lower level (page or record) when one transaction requests an X lock.

5. (3 points) Mark all the transactions that are involved in a deadlock. If there is no deadlock, mark No Deadlock.

$T_1$	X(A)									S(D)
$T_2$		S(A)								
$T_3$			X(B)					S(C)	X(A)	
$T_4$				S(C)			X(A)			
$T_5$					S(D)	X(B)				

- A.  $T_1$   
 B.  $T_2$   
 C.  $T_3$   
 D.  $T_4$   
 E.  $T_5$   
 F. No Deadlock
6. (1 point) Which of the following applies for the “wait-die” deadlock avoidance strategy?  
 A. This deadlock avoidance strategy avoids all deadlocks.  
 B. This deadlock avoidance strategy never aborts unnecessarily.
7. (1 point) Which of the following applies for the “wound-wait” deadlock avoidance strategy?  
 A. This deadlock avoidance strategy avoids all deadlocks.  
 B. This deadlock avoidance strategy never aborts unnecessarily.
8. (3 points) Consider a “wait-wait-die” deadlock avoidance strategy, where when  $T_i$  requests a lock that  $T_j$  holds, it follows the following protocol:
- if  $T_j$  has higher priority,  $T_i$  waits,
  - else if no transaction is waiting on  $T_i$ ,  $T_i$  waits,
  - else,  $T_i$  aborts.

Which of the following applies for the “wait-wait-die” deadlock avoidance strategy?

- A. This deadlock avoidance strategy avoids all deadlocks.  
 B. This deadlock avoidance strategy never aborts unnecessarily.  
 C. This deadlock avoidance strategy aborts less frequently than the wait-die strategy.

### 3 Recovery (14 points)

1. (4 points) For each of the following four questions, mark True or False.
  - A. A NO STEAL policy is useful for achieving atomicity without REDO logging.
  - B. A FORCE policy is useful for achieving durability without REDO logging.
  - C. An implementation of the REDO phase that only checks  $\text{pageLSN} \geq \text{LSN}$  and not the other 2 criteria is an incorrect implementation (here correctness means that an operation is redone in this implementation if and only if it is redone in the original implementation from lecture).
  - D. Assume that a system only allows at most  $n$  active transactions. In the worst case, there is no ACID-preserving mechanism to bound the number of operations undone in the undo phase by some constant  $k$ .

Consider the following log recovered after a crash. Assume the buffer pool flushes to disk all pages referenced up to (and including) LSN 60, and does not flush any pages after that.

LSN	Record	prevLSN
70	UPDATE T2 W(P3)	null
80	UPDATE T1 W(P1)	null
90	Begin Checkpoint	—
100	UPDATE T3 W(P2)	null
110	UPDATE T1 W(P2)	80
120	End Checkpoint	—
130	COMMIT T2	70
140	COMMIT T3	100
150	END T2	130
160	UPDATE T4 W(P1)	null
170	UPDATE T1 W(P1)	110
180	ABORT T1	170

2. (5.5 points) Fill out what the DPT and Xact table look like at the beginning of the REDO phase (after ANALYSIS ends). Please write **None** for any fields that should not be in the tables. For the Xact table Status, the possible options are: running, committing, aborting.

PID	recLSN
1	(a)
2	(b)
3	(c)

TID	lastLSN	Status
1	(d)	(e)
2	(f)	(g)
3	(h)	(i)
4	(j)	(k)

3. (3 points) For each of the following questions, mark True if the record is added to the log after recovery, and False otherwise.
  - A. A CLR for T2's update at LSN 70
  - B. A CLR for T1's update at LSN 80
  - C. A CLR for T3's update at LSN 100
  - D. A CLR for T1's update at LSN 110
  - E. A CLR for T4's update at LSN 160
  - F. A CLR for T1's update at LSN 170

4. (1.5 points) Mark the box corresponding to the order of transactions (ascending by LSN) in which the CLR records are added. For example, if we had a log that looked like this

LSN	Record	prevLSN
190	CLR T5 ...	...
200	CLR T6 ...	...
210	CLR T7 ...	...

The ordering would be

T5, T6, T7

- A. T1, T1, T1, T4, T3, T2
- B. T1, T4, T1, T3, T1, T2
- C. T1, T1, T1, T4
- D. T1, T4, T1, T1
- E. T1, T1, T1, T3, T2
- F. T1, T1, T3, T1, T2

## 4 Text Search Incl Ranking (TF-IDF) (12 points)

- (6 points) For each assertion, fill in the corresponding bubble True or False.
  - In the “Bag of Words” model, we might choose to ignore the word “of” because it doesn’t contain much information. This is an example of a stop word.
  - To allow fast text searches, we build inverted files, which contain maps from documents to words.
  - The vector space of an IR system is enormous because it has one “dimension” per document in the corpus.
  - When calculating distance to find similarity between docs, it is beneficial to normalize the vectors.
  - Consider three docs: Doc1 has length 10,000, Doc2 has length 9,900, Doc3 has length 100. If we use Cosine Similarity Distance Metric, Doc1 is very likely to be more similar to Doc2 than to Doc3.
  - TF-IDF value increases as the number of times a word appears in the document increases.
- (1 point) When doing IR Ranking, what types of words do we favor? **There may be zero, one, or more than one correct answer.**
  - repeated words
  - long words
  - unusual words
  - capitalized words
- (2 points) Consider 500 documents with docID  $1, \dots, 500$ . The following table contains the number of occurrences of the word “database” in each document:

docID	number of “database”
1	4
2	17
3	12
4	7
5	9
6-500	0

What is the “DocTermRank” for term = “database” and docID = 2? (Hint: consider TF and IDF). Assume base-10 for any logarithms.

4. (1 point) We run a text search query and get the top 100 answers. We find that 95 of them are correct. We say that this is a good answer because 95% of them are correct. Which metric of answer quality is this?
- A. Accuracy
  - B. Precision
  - C. Retrieval
  - D. Recall

For the next two problems, consider the following expressions:

- A.  $(\# \text{ of true positives}) / ((\# \text{ of true positives}) + (\# \text{ of false positives}))$
  - B.  $(\# \text{ of true positives}) / ((\# \text{ of true positives}) + (\# \text{ of true negatives}))$
  - C.  $(\# \text{ of true positives}) / ((\# \text{ of true positives}) + (\# \text{ of false negatives}))$
  - D.  $(\# \text{ of true positives}) / (\# \text{ of true negatives})$
5. (1 point) Which of the above 4 expressions above is a definition of precision?
6. (1 point) Which of the above 4 expressions above is a definition of recall?

## 5 Distributed Transactions with Two Phase Commit (2PC) (10 points)

1. (4 points) For each of the following four questions, mark True or False.
  - A. Distributed deadlock occurs only if the waits-for graph at a node forms a cycle.
  - B. In 2PC, we need all participant nodes to agree on abort (vote NO) to abort a transaction.
  - C. A participant machine that does not flush its prepare records to the log before responding to the coordinator could violate the durability property.
  - D. In 2PC phase 2, after the coordinator sends commit message to all participants, participants first respond with Ack, and then generate and flush commit record.
  
2. (2 points) Suppose you are you are a machine that has just recovered from a crash. You discover only an abort record in your log for transaction T. Under proper 2PC and logging protocols, mark True or False for each of the following two questions.
  - A. You are a participant.
  - B. You crashed during phase 2.
  
3. (2 points) Suppose you are a participant machine that has just recovered from a crash. There is only one record in your log for transaction T. Under proper 2PC and logging protocols, mark True or False for each of the following two questions.
  - A. You know all the messages you received before you crashed.
  - B. You need to decide if you need to commit or abort transaction T.
  
4. (2 points) Suppose in 2PC with logging, we have one coordinator and three participants. It takes 30ms for a coordinator to send messages to all participants; 5, 10, and 15ms for participant 1, 2, and 3 to send a message to the coordinator respectively; and 10ms for each machine to generate and flush a record. Assume for the same message, each participant receives it from the coordinator at the same time.  
Under proper 2PC and logging protocols, how long does the whole 2PC process (from the beginning to the coordinator's final log flush) take for a successful commit in the best case?

## 6 SQL/Relational Algebra (10.5 points)

Use your knowledge of SQL to deduce what is happening during the Battle of Hogwarts. You may assume you have the following tables:

```
CREATE TABLE Wizards(wizid integer, name text, house text, evil boolean, PRIMARY KEY(wizid));
```

```
CREATE TABLE Spells(sid integer, name text, offensive boolean, PRIMARY KEY (sid));
```

```
CREATE TABLE Attacks(attackid integer, attacker integer, attacked integer, spell integer,
PRIMARY KEY (attackid), FOREIGN KEY(spell) REFERENCES Spells,
FOREIGN KEY(attacker) REFERENCES Wizards, FOREIGN KEY(attacked) REFERENCES Wizards);
```

**Disclaimer:** For all of the following questions, you do not need any Harry Potter knowledge. Any understanding of Wizards or Spells will not be helpful.

1. (1.5 points) Select all of the following queries that return the name of each wizard who has been an attacker more than 3 times. Do not assume that names are unique.

A. 

```
SELECT name FROM Wizards, Attacks
WHERE wizid = attacker
GROUP BY attacker, name
HAVING COUNT(*) > 3;
```

B. 

```
SELECT name FROM Wizards, Attacks
WHERE wizid = attacker
GROUP BY name
HAVING COUNT(*) > 3;
```

C. 

```
SELECT name FROM
(SELECT name FROM Wizards, Attacks
WHERE wizid = attacker) AS a
GROUP BY attacker, name
HAVING COUNT(*) > 3
ORDER BY COUNT(name);
```

2. (1.5 points) Select all of the following queries that select the names of wizards (A) that another individual wizard (B) attacked twice, where the attacker (B) used two different spells. There should be no duplicates in this list. Do not assume that names are unique.

A. 

```
SELECT w1.name AS A
FROM Wizards w1, Wizards w2, Attacks a1, Attacks a2
WHERE w1.wizid = a1.attacked AND a1.spell <> a2.spell
AND w2.wizid = a2.attacked AND a1.attacked = a2.attacked;
```

B. 

```
SELECT DISTINCT w1. name AS A
FROM Wizards w1, Attacks a1, Attacks a2
WHERE w1.wizid = a1.attacked AND a1.spell <> a2.spell
AND w1.wizid = a2.attacked AND a1.attacked = a2.attacked;
```

C. 

```
SELECT DISTINCT w1.name AS A
FROM Wizards w1, Wizards w2, Attacks a1, Attacks a2
WHERE a1.attacker = a2.attacker AND w1.wizid = a1.attacked
AND a1.spell <> a2.spell AND w2.wizid = a2.attacked AND a1.attacked = a2.attacked;
```

3. (1.5 points) Select all of the following queries that return the name of all of the spells that have never been used in an attack.

- A. 

```
SELECT name
FROM Spells
WHERE sid NOT IN
(SELECT spell FROM Attacks);
```
- B. 

```
SELECT name
FROM Spells
WHERE NOT EXISTS
(SELECT * FROM Attacks WHERE spell = sid);
```
- C. 

```
SELECT Spells.name
FROM Spells, Wizards, Attacks
WHERE wizid = attacker and spell = sid;
```

For the following questions: fill in the blanks to create a query that returns true if more evil wizards cast offensive spells, and returns false if more good wizards cast offensive spells.

Note that **exactly one** answer will be correct.

```
WITH count_evil_good_spells(evil, num_offensive) AS
(SELECT evil, count(*)
FROM attacks, wizards, spells
WHERE ___(4)___ and spell = sid
AND offensive = ____ (5) ____
GROUP BY ____ (6) ____)
SELECT evil
FROM count_evil_good_spells
WHERE num_offensive >= ALL
(SELECT num_offensive
FROM ___ (7) ___);
```

4. (0.5 points) Fill in the blank labeled (4)

- A. `wizid=attackid`
- B. `wizid=attacked`
- C. `wizid=attacker`

5. (0.5 points) Fill in the blank labeled (5)

- A. `true`
- B. `false`

6. (0.5 points) Fill in the blank labeled (6)

- A. `evil`
- B. `spell`
- C. `attacker`

7. (0.5 points) Fill in the blank labeled (7)

- A. `Wizards`
- B. `count_evil_good_spells`
- C. `Attacks`

8. (2 points) Select all of the following answers that return the **wizid** of all of the wizards who have not attacked anybody.

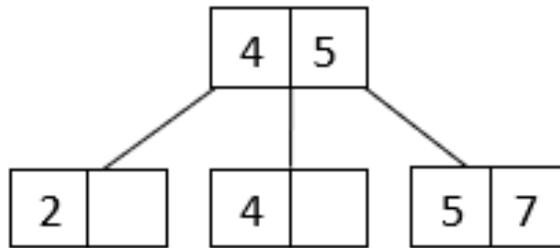
- A.  $\pi_{wizid}(\text{Attacks} - \text{Wizards})$
- B.  $\pi_{wizid}(\pi_{wizid}(\text{Wizards}) - \pi_{attacker}(\text{Attacks}))$
- C.  $\pi_{wizid}(\text{Wizards}) - \pi_{attacker}(\text{Spells} \bowtie_{sid = spell} \text{Attacks} \bowtie_{attacker = wizid} \text{Wizards})$
- D.  $\pi_{wizid, name}(\text{Wizards}) - \pi_{attacker}(\text{Attacks})$

9. (2 points) Select all of the following answers that return the **wizid** of all of the wizards who were attacked by Wizards whose house is Gryffindor.

- A.  $\pi_{attacked}(\text{Spells} \bowtie_{sid = spell} \text{Attacks} \bowtie_{attacker = wizid} \sigma_{house = 'Gryffindor'}(\text{Wizards}))$
- B.  $\pi_{attacked}(\text{Spells} \bowtie_{sid = spell} \text{Attacks} \bowtie_{attacker = wizid} \sigma_{house = 'Gryffindor'}(\pi_{wizid}(\text{Wizards})))$
- C.  $\pi_{wizid}(\text{Wizards}) - \pi_{attacked}(\text{Spells} \bowtie_{sid = spell} \text{Attacks} \bowtie_{attacked = wizid} \sigma_{house \neq 'Gryffindor'}(\text{Wizards}))$
- D.  $\pi_{attacked}(\text{Attacks} \bowtie_{attacker = wizid} (\sigma_{house = 'Gryffindor'}(\text{Wizards}) \bowtie_{sid = spell} \text{Spells}))$

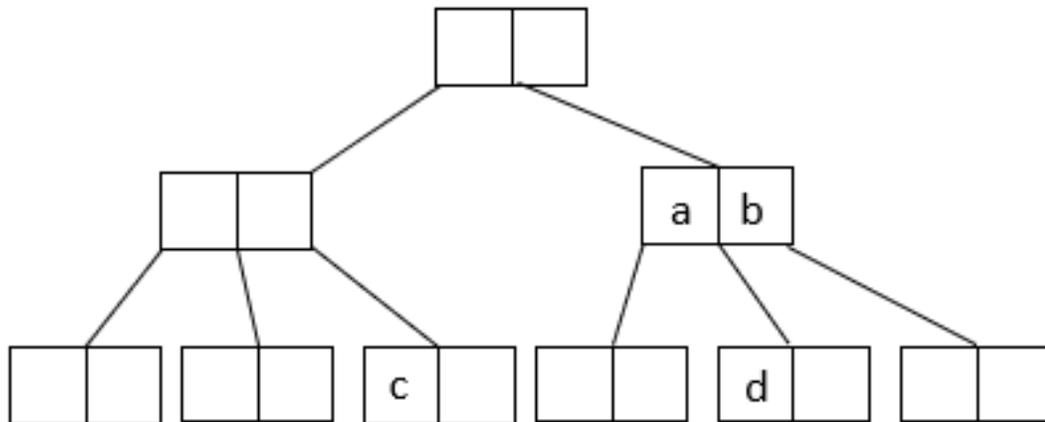
## 7 B+ Trees (15 points)

1. (5 points) For each of the following five questions, mark True or False.
  - A. You cannot build a B+ tree on a variable length column such as a string.
  - B. Indexes are used to optimize tables that are written to much more frequently than they are read from.
  - C. For alternative 2 trees, clustered indexes will not save any IOs over unclustered indexes when you are only doing equality search.
  - D. Alternative 1 indexes will usually be able to store fewer records in each leaf than Alternative 2 indexes.
  - E. With a series of carefully constructed inserts, an adversary can cause a B+ tree to be unbalanced, so that some lookups visit more nodes than others.
2. (4 points) Given the following B+ tree:



What will the B+ tree look like after inserting 1, 3, and 6 in that order? Fill in the answer sheet with what number goes in boxes a-d using the following template. Put none if the box is empty. Some pages in the template may not have any entries on in them.

**Reminder:** when a node splits and there is an odd number of entries, the majority goes to the right node.





## 8 Query Processing and Parallel DBMS (10 points)

1. (2 points) For each of the following three questions, mark True or False.
  - A. Performing an index nested loop join with a clustered index on the inner relation always requires more I/Os than an index nested loop join with an unclustered index on the inner relation (for the same tables).
  - B. Pipeline parallelism scales up with the amount of data.
  - C. Performing key lookup over hash partitioned data requires fewer I/Os than if the data had been round-robin partitioned.

We want to join together the Owners and Pets table on petID so we can later on compare the owner's happiness to the pet's happiness. Suppose we have the following tables and variables:

Owners: (ownerID, name, happiness, petID)

Pets: (petID, happiness, ownerID)

variable	symbol	value
pages of Owners table	$[O]$	30
tuples per Owners page	$p_O$	5
pages of Pets table	$[P]$	45
tuples per Pets page	$p_P$	5
pages in memory per machine to perform the join	B	6
Number of machines	M	3
Size of Page	s	2KB

**Note:** For these questions, do **NOT** include the cost of writing matching output, but **DO** include the cost of scanning the tables.

Assume that we have **three** machines. We want to join Owners and Pets on Owners.petID = Pets.petID.

2. (2 points) If the data starts out round-robin partitioned, what is the total amount of data in **KB** transmitted over the network during repartitioning in order to perform **grace hash join**?
3. (2 points) Assume that we can partition and perform the initial pass of hashing at the same time. Assuming that the data starts out round-robin partitioned across the machines, how many (disk) I/Os are needed **in total across all machines** to perform **grace hash join**?
4. (2 points) Assume that the data starts out partitioned perfectly across the machines for a **sort-merge join**, and no network communication is required. How many (disk) I/Os are needed **per machine** to perform sort merge join, using the optimization if possible? (Note: your answer should be per machine, in contrast to the previous question.)
5. (2 points) How many (disk) I/Os would be needed to perform sort merge join using the optimization if possible if we only had **one** machine and **6 buffer pages** for this one machine instead?

## 9 Query Optimization (16 points)

A relational database has the table:

```
CREATE TABLE Employee(PNR text, SALARY integer, SEX text, DEPT text, PRIMARY KEY(PNR))
```

The cost-based optimizer uses the following statistics stored as meta-data in the DBMS's system catalogue:

- There are 10000 rows in the table.
- There are 20 departments in the company.
- SEX is either 'Male' or 'Female'.
- The lowest salary is 20000 and the highest is 100000.

The query optimizer make the following assumptions on statistics:

- The query optimizer assumes even distributions of data values in columns
- The query optimizer assumes independence between values in different columns

For questions of selectivity, you may answer either with a fraction or a decimal.

1. (2 points) What is the selectivity of the condition: `SALARY < 25000`?
2. (2 points) What is the selectivity of the condition: `(SALARY > 100000 OR SEX != 'Male')` And `DEPT='Toys'`?
3. (2 points) What is the selectivity of the condition: `(SEX = 'Male' OR SALARY > 80000)`?
4. (2 points) What is the selectivity of the condition: `((SEX= 'Male' AND DEPT = 'Toys') OR (SEX='Female' AND DEPT = 'Tools')) AND (SALARY > 50000 OR SALARY < 25000)`?

Consider the following SQL query that finds all applicants who want to major in CS, live in Berkeley, and go to a school ranked better than 10 (i.e., rank < 10).

Relation	Cardinality	Number of pages	Primary key
Applicants ( <u>id</u> , name, city, sid)	2,000	100	id
Schools ( <u>sid</u> , sname, srank)	100	10	sid
Major ( <u>id</u> , major)	3,000	200	(id,major)

```

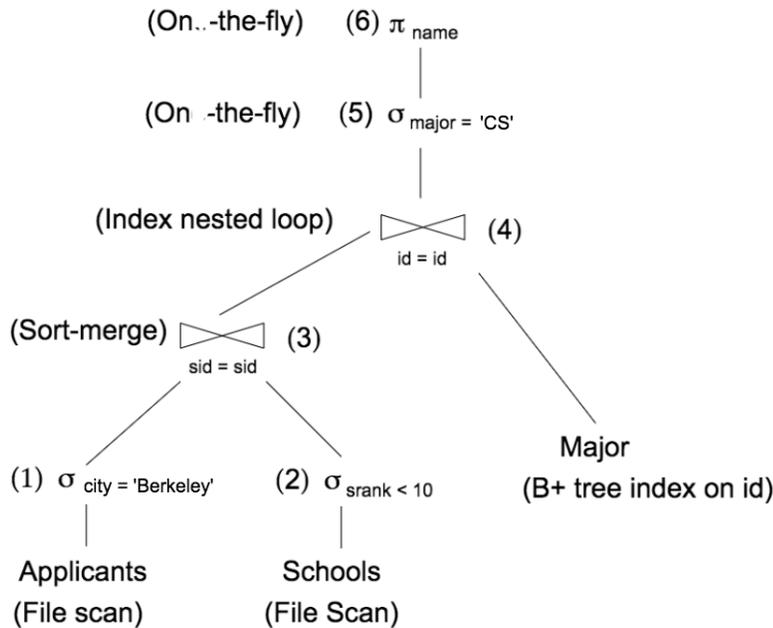
SELECT A.name
FROM Applicants A, Schools S, Major M
WHERE A.sid = S.sid AND A.id = M.id
AND A.city = 'Berkeley' AND S.srank < 10 AND M.major = 'CS'

```

Assume that:

- Each school has a unique rank number (srank value) between 1 and 100 (both inclusive).
- There are 20 different cities.
- Applicants.sid is a foreign key that references Schools.sid.
- Major.id is a foreign key that references Applicants.id.
- There is an unclustered, alternative 2 B+ tree index on Major.id and all index pages (a total of 5) are already in memory.
- For Major table, all tuples belonging to same student are stored in one physical page.
- The buffer size is 150 pages.

The query optimizer is currently considering the below query plan.



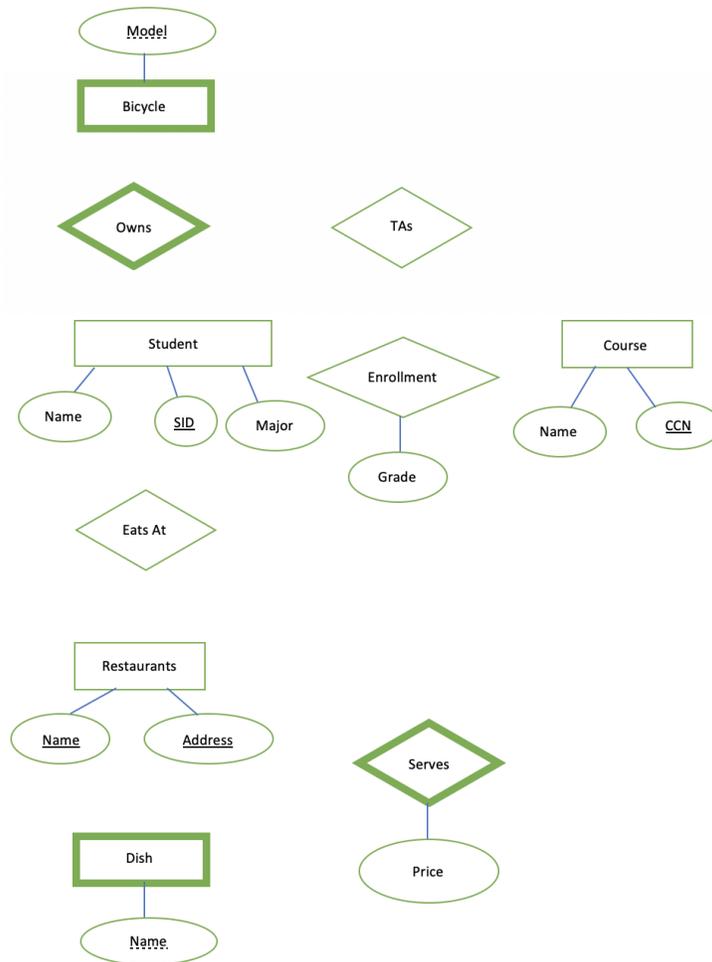
5. (3 points) What is the cardinality of the output of operator 1 in unit of tuples?

6. (5 points) Suppose the output cardinality of operator 3 is known to be 9 tuples and they have different Applicant ids. What is the total I/O cost of this whole plan? Do not include the cost of writing the final result.

## 10 ER Diagrams (10 points)

For questions 1-10, you will fill in the following ER Diagram, which models parts of a student's routine in a University. (Hint: You might want to fill the diagram while you read these requirements here). The constraints are listed below.

- Students own at least one bicycle. Each bicycle is uniquely identified by its owner and the bicycle model.
- Students must take at least one course, and every course must have at least one student.
- Students may TA for at most one course, but are under no obligation to teach at all. Every course can have as many TAs as it needs, or none at all.
- Students can choose to eat at restaurants, but are under no obligation to do so. Every restaurant must have some students eating at it, however.
- Each restaurant, identified by its address and name, has to serve multiple dishes. Each dish is uniquely identified by its name and the restaurant serving it.



For each of the following questions, mark **one** of the following choices:

- A. Thin Arrow
- B. Thick Arrow
- C. Thin Line
- D. Thick Line

1. (1 point) Which edge should we draw to connect the **Students** entity with the **Enrollment** relationship set?
2. (1 point) Which edge should we draw to connect the **Course** entity with the **Enrollment** relationship set?
3. (1 point) Which edge should we draw to connect the **Students** entity with the **TAs** relationship set?
4. (1 point) Which edge should we draw to connect the **Course** entity with the **TAs** relationship set?
5. (1 point) Which edge should we draw to connect the **Student** entity with the **Owns** relationship set?
6. (1 point) Which edge should we draw to connect the **Bicycle** entity with the **Owns** relationship set?
7. (1 point) Which edge should we draw to connect the **Restaurants** entity with the **Eats At** relationship set?
8. (1 point) Which edge should we draw to connect the **Student** entity with the **Eats At** relationship set?
9. (1 point) Which edge should we draw to connect the **Restaurants** entity with the **Serves** relationship set?
10. (1 point) Which edge should we draw to connect the **Dish** entity with the **Serves** relationship set?