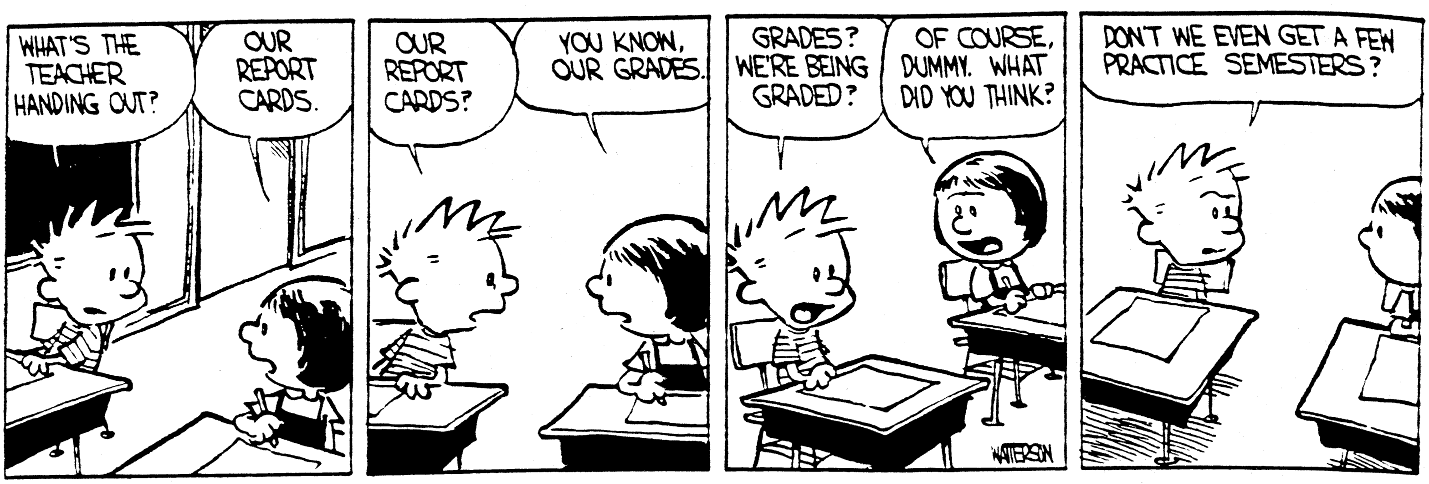
CS 564 Final Exam, Fall 2013

Please **print** your name below

Last Name

First Name

Instructions:
1. This is a closed book exam.
2. You have 120 minutes to complete this exam. The points on this exam total to 120.



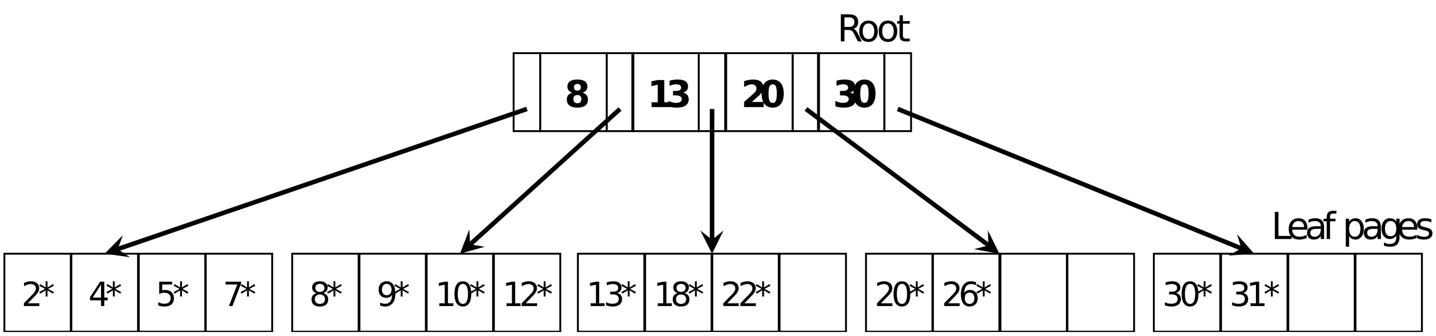
For Instructor's use only

Question 1: Indexing and Sorting (30 points)	
Question 2: RA, SQL, and ER (35 points)	
Question 3: Query Evaluation and Optimization (30 points)	
Question 4: Normalization and Concurrency (25 points)	
TOTAL	

Question 1: [30 points] Indexing and Sorting

a) **[6 points]** Consider a B+-tree with a maximum fanout of 64 for both the leaf and the non-leaf nodes. If you are told that the average occupancy of each node in this tree is 50%, then calculate the number of entries in the index for a tree of height 3, i.e. there is root level, two non-leaf levels below the root, and then the leaf level.

b) **[4 points]** Draw the final B+-tree after inserting a key with value 11 in the B+-tree shown below. Assume that the insertion algorithm does not allow any redistribution. The order of the tree below is 2.



c) **[12 points]** Imagine using external sort to sort a file with 1024 pages, and using a buffer of 64 pages. In the first pass, you use replacement sort to produce sorted runs. Then, in the subsequent merge passes you use blocked I/Os to read 4 pages at a time.

i. [1 point] What is the average size of each run at the end of the first pass?

ii. [5 points] Following the first pass, how many passes are needed to produce the fully sorted file? Show how you derive your answer.

iii. [6 points] Instead of reading 4 pages at a time above, if you read 2 pages at a time and used double buffering, how would your answer above in (ii) change. Show how you derive your answer.

d) **[8 points]** Consider a composite B+-tree index on attributes (a, b, c). For each of the predicate below answer the following question: Can you use this index to efficiently answer the predicate that is shown? <u>If the answer is no</u>, explain briefly. No points will be awarded if the justification is incorrect.

a < 10 and b = 5 ☐ Yes ☐ No

b = 10 and c = 5 ☐ Yes ☐ No

a = 10 and c = 5 ☐ Yes ☐ No

a > 10 and b < 10 and c > 5 ☐ Yes ☐ No

Question 2: [35 points] RA, SQL and ER

a) **[20 points]** Consider the schema for the three relations below. The primary keys are marked in bold, and the foreign keys are underlined. So, <u>start-city</u> is a foreign key that references the primary key "cid" in the City relation. Similarly <u>end-city</u> is also a foreign key. And, <u>*aid*</u> is a foreign key pointing to Airline.

City (**cid**, name, population); Flight (**fid**, length, <u>start-city</u>, <u>end-city</u>, <u>aid</u>); Airline (**aid**, name, profit);

i. **[4 points]** Write the following query in **Relational Algebra**. Find the flight *fid(s)* for flights that start in a city with id "MSN" and end in a city with "LON".

ii. **[6 points]** Write the following query in **Relational Algebra**. Find the *names* of the cities that have a flight for every airline with profit more than 0.

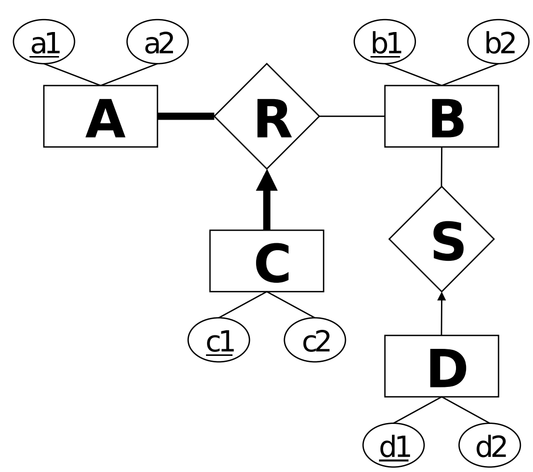
Recall the schema that was introduced on the previous page. Here it is again, so you don't have to flip pages.

City (**cid**, name, population); Flight (**fid**, length, start-city, end-city, aid);
Airline (**aid**, name, profit);

iii. **[4 points]** Write the following query in **SQL**: Print the name of each city that has some flight starting from it by an airline that has profit more than 0. (So only print cities that have flights starting from it by airlines that make a profit.)

iv. **[6 points]** Write the following query in **SQL**: Print the expression "population divided by the number of flights that start from that city" for every city that has population greater than 250,000.

b) **[15 points]** Translate the ER diagram shown in the adjacent figure into a relational schema by writing the SQL CREATE TABLE statements to create the database. Make sure you clearly mark all the keys in the relations. You can assume that all attributes are of type Integer.



CREATE TABLE A (a1 INTEGER PRIMARY KEY, a2 INTEGER);

CREATE TABLE RC(c1 INTEGER PRIMARY KEY,

```
                    c2 INTEGER,
                    a1 INTEGER NOT NULL,
                    FOREIGN KEY (a1) REFERENCES A)

CREATE ASSERTION ARConstraint (SELECT COUNT( DISTINCT a1) FROM RC = (SELECT
COUNT(a1) FROM A)

CREATE TABLE B(b1 INTEGER PRIMARK KEY, a2 INTEGER);

CREATE TABLE SD(    d1 INTEGER PRIMARY KEY,
                    d2 INTEGER,
                    b1 INTEGER,
                    FOREIGN KEY (b1) REFERENCES B)
```

Question 3: [30 points] Query Evaluation and Optimization

a) **[15 points]** Consider computing an equijoin between two relations R and S, with 100 and 1000 pages respectively; i.e. |R| = 100 and |S| = 1000. Assume a buffer pool of 10 pages. Mark True or False below, and provide a brief explanation for your answer. No points will be awarded if the justification is incorrect.

The block nested loops join algorithm **will always** be faster than page nested loops join. (Recall that page nested loops is a simple modification of the tuple nested loops join algorithm but works on a page of tuples at a time.)	☐ True ☐ False	Justification:
The sort-merge join algorithm **will always** be faster than a hash-based join algorithm in this case.	☐ True ☐ False	Justification:
If there is an index built of the join attribute of S, then a index nested loops join algorithm (in which one probes the S index for	☐ True ☐ False	Justification:

every tuple of R), will **always be faster** than block nested loops.		

b) **[15 points]** Consider a DBMS system that has an implementation for Sort-merge Join and Block Nested Loops join algorithms. Now consider optimizing the following SQL query:

```
SELECT *
FROM R, S, T
WHERE R.a = S.a
    AND R.a = T.b
    AND R.a = R.d
```

There is a B-tree index on R.a, another on R.d, another on S.a, and the last one on T.z (T.z is not referenced in the query)

- **[3 points]** What is the <u>maximum</u> number of single relation-access plans **<u>for R</u>** that the optimizer will retain at the end of the first phase? Explain your answer.

Your Answer: The maximum number of single relation plans are: ____________________

The plans are:

- **[3 points]** What is the <u>maximum</u> number of single relation-access plans **<u>for T</u>** that the optimizer will retain at the end of the first phase? Explain your answer.

Your Answer: The maximum number of single relation plans are: ____________________

The plans are:

- **[4 points]** List all the interesting attributes in the query.

- **[5 points]** Assume that the optimizer retains only one single relation access plan for R and T, and two for S. How many two-relation access plans will the optimizer consider in the next phase?

Your Answer: The maximum number of two-relation plans are: ____________________

The plans are:

Question 4: [25 points] Normalization, and Concurrency

a) **[5 points]** Consider the following relational schema for a relation R with four attributes: R(a, b, c, d). This relation has the following three functional dependencies:

$a \rightarrow b$

$b \rightarrow \{a,d\}$

$b \rightarrow c$

From these functional dependencies what can you say something about the property of the attribute a. Explain.

b) **[8 points]** Consider using normalization to decompose a relational schema with redundancies. Mark True or False below and provide a brief explanation to justify your answer. No points will be awarded if the justification is not correct.

It is okay for the decomposed relations to **not be dependency preserving**	☐ True ☐ False	Justification:
It is okay for the decomposed relations to **not be "lossless join"**	☐ True ☐ False	Justification:

c) **[12 points]** This question is about concurrency control. Mark True or false below and provide a brief explanation to justify your answer. No points will be awarded if the justification is not correct.

For any schedule, there is only one acceptable schedule. (Recall that an acceptable schedule is on that has the same outcome as some serial schedule.)	☐ True ☐ False	Justification:

When using locking for concurrency control, using two phase locking guarantees serializability.	☐ True ☐ False	Justification:
Deadlocks are an indication that the “isolation” property has been violated. (Recall that locking is a mechanism for concurrency control, and concurrency control provides the “I” in ACID, where “I” stands for isolation.	☐ True ☐ False	Justification:

Extra Credit Question (2 points): What is the difference between 2PL (Two-phase locking) and strict 2PL?