Name

CWID

Exam
2

Dec 7th, 2020

CS525 - Final Exam
Solutions

Please leave this empty! 1 2 3 4 5 6 7

Sum

Instructions

- The exam is from 6am to 6pm and **open books and open notes**
- For your convenience the number of points for each part and questions are shown in parenthesis.
- There are 7 parts in this exam (100 points total)
    1. SQL (26)
    2. Relational Algebra (15)
    3. Index Structures (20)
    4. I/O Estimation (12)
    5. Result Size Estimation (12)
    6. Schedules (15)

For this exam, I state truthfully: I have not received, I have not given, nor will I give or receive, any assistance to another student taking this exam, including discussing the exam with students in another section of the course, or using any online service designed to share information about exams.

Student signature

Part 1 SQL (Total: 26 Points)

Consider the following database storing information about a company's warehouses, orders, pricing, and stock.

warehouse

location	size	num_employee
Chicago	100,000	200
Schaumberg	20,000	15
Michigan City	30,000	20

distance

from	to	distance
Chicago	Schaumberg	15.0
Schaumberg	Chicago	15.0
Chicago	Michigan City	30.5
Michigan City	Chicago	30.5
Schaumberg	Michigan City	44.0
Michigan City	Schaumberg	44.0

order

oid	location	item	cnt
1	Chicago	Shovel	2
2	Chicago	Lawnmower	1
3	Schaumberg	Lawnmower	1
4	Michigan City	Lawnmower	20

price

location	item	price
Chicago	Shovel	50
Schaumberg	Shovel	45
Michigan City	Shovel	35
Chicago	Lawnmower	250
Schaumberg	Lawnmower	225
Michigan City	Lawnmower	260

stock

location	item	stockcnt
Chicago	Lawnmower	53
Michigan City	Lawnmower	10
Schaumberg	Shovel	25
Chicago	Shovel	1

Hints:

- When writing queries do only take the schema into account and **not** the example data given here. That is your queries should return correct results for all potential instances of this schema.
- Attributes with black background form the primary key of an relation. For example, **location** is the primary key of relation **warehouse**.
- The attributes **from** and **to** of relation **distance** are foreign keys to relation **warehouse**.
- The attributes **location** of relation **order** is a foreign key to relation **warehouse**.
- The attributes **location** of relation **price** is a foreign key to relation **warehouse**.
- The attributes **location** of relation **stock** is a foreign key to relation **warehouse**.

Question 1.1 (4 Points)

Write a SQL query that computes the total price of all items in stock for each location and returns the 3 locations with the highest total stock value (total price of all items in stock at that location). Note that prices are specific to locations.

Solution

```
SELECT s.location, sum(price * stockcnt) AS ttlprice
FROM stock s, price p
WHERE s.location = p.location AND s.item = p.item
GROUP BY s.location
ORDER BY ttlprice DESC
LIMIT 3;
```

Question 1.2 (4 Points)

Write a SQL query that returns the cheapest location to fulfill the following two orders (both orders should be fulfilled at the same location):

- 3 Shovels
- 15 Lawnmowers

Solution

```
SELECT location, sum(CASE WHEN item = 'Shovel' THEN 3 * price
                          WHEN item = 'Lawnmower' THEN 15 * price
                          ELSE 0 END) AS ttlprice
FROM price
GROUP BY location
ORDER BY ttlprice ASC
LIMIT 1;
```

Question 1.3 (6 Points)

Write a SQL statement that returns the locations for which all orders can be served locally. For instance, in the example database, the Chicago orders cannot be served locally from the Chicago warehouse, because there are not enough shovels in stock to fulfill the first order (even though the order with oid 2 can be fulfilled, because there are enough lawnmowers in stock in Chicago).

Solution

```
WITH requirements AS (
  SELECT location, item, sum(cnt) AS req
  FROM order
  GROUP BY location, item)
SELECT location
FROM (SELECT location, min(CASE WHEN s.stockcnt >= r.req THEN 1 ELSE 0) AS enough
      FROM requirements r
           LEFT OUTER JOIN
           stock s ON (r.location = s.location AND r.item = s.item)
      GROUP BY location)
WHERE enough = 1;
```

Question 1.4 (6 Points)

Write a SQL query that returns the locations at which all the orders can be fulfilled simultaneously. A location can fulfill all of the orders, if it has enough items in stock to fulfill all of the orders (the sum of `cnt` for each item type is less than or equal to the `stockcnt` for this item type at this location).

Solution

```
WITH totitems AS (
  SELECT sum(cnt) AS ttlcnt, item
  FROM order
  GROUP BY item),
notenough AS (
  SELECT DISTINCT location
  FROM (warehouse w CROSS JOIN totitems t)
       LEFT OUTER JOIN
       stock s ON (t.item = s.item AND w.location = t.location)
  WHERE s.stockcnt IS NULL OR s.stockcnt < ttlcnt)
SELECT location
FROM warehouse
WHERE location NOT IN (SELECT * FROM notenough);
```

Question 1.5 (6 Points)

Write a SQL query that returns for each order the price of the cheapest option to fulfill this order. To fulfill an order we can use a combination of items that are available at the warehouse at the location of the order and / or can ship items (if enough are available) from other locations. The price payed for an item is the price at the location from where we ship the item. When shipping an item from a location X to a location Y, we have to pay a cost for shipping that is calculated as the `number of items` times `the distance between X and Y` times `0.2`.

For example, in the example database one option for fulfilling the order `(Chicago,Shovel,2)` is to:

- take the only shovel from the local warehouse in Chicago ($50)
- ship one shovel from Schaumberg ($45 per shovel) to Chicago (shipping cost is $1 \times \$15.0 \times 0.2 = \$3$).

The total cost of this option would be $\$50 + \$45 + \$3 = \$98$. The only other option in this example is to ship both shovels from Schaumberg $2 \times \$48 = \$96$. Thus, in this case both Shovels should be shipped from Schaumberg.

This question is significantly harder than the other questions in this exam. Maybe solve this at the end.

Solution

```
WITH locavail AS (
  SELECT oid, cnt, o.location AS oloc, s.location AS sloc, price,
         LEAST(stockcnt,cnt) AS avail
  FROM order o, stock s, price p
  WHERE o.item = s.item AND s.location = p.location AND s.item = p.item),
numtotake AS (
  SELECT oid, oloc, sloc, price,
         (CASE WHEN takecur < cnt THEN avail
               ELSE cnt - takeprev END) AS take
  FROM (SELECT oid, oloc, sloc, price, avail,
               sum(avail) OVER (PARTITION BY oid ORDER BY price) AS takecur,
               sum(avail) OVER (PARTITION BY oid ORDER BY price
                                ROWS BETWEEN UNBOUNDED PRECEDING
                                     AND 1 PRECEDING) AS takeprev
        FROM locavail) curandprevtake)
SELECT oid, sum(take * price + CASE WHEN oloc != sloc
                                    THEN distance * 0.2
                                    ELSE 0 END) AS cheapest_price
FROM numtotake n, distance d
WHERE d.from = sloc AND d.to = oloc
GROUP BY oid;
```

Part 2 Relational Algebra (Total: 15 Points)

Question 2.1 Relational Algebra (5 Points)

Write a relational algebra expression over the schema from the SQL part that returns the pair of locations with the shortest distance between each other. Use the **bag semantics** version of relational algebra.

Solution

$$\pi_{from,to}(\ \alpha_{min(distance)}(distance) \bowtie_{min(distance)=distance}\ distance)$$

Question 2.2 Relational Algebra (5 Points)

Write a relational algebra expression over the schema from the SQL part that returns for each order the locations at which the order can be fulfilled (there are enough items in stock to fulfill the order). Use the **bag semantics** version of relational algebra.

Solution

$$\pi_{oid,location}(\pi_{oid,item,cnt}(order) \bowtie_{item=item \wedge stockcnt \geq cnt}\ stock)$$

Question 2.3 Relational Algebra (5 Points)

Write a relational algebra expression over the schema from the SQL part that returns the total cost of fulfilling all orders locally (at the location where the order was submitted, i.e., `order.location`). Only return order where there is enough local stock to fulfill the order locally. Use the **bag semantics** version of relational algebra.

Solution

$$\pi_{oid,price*cnt}((\pi_{oid,location,item,cnt}(order \bowtie_{item=item \wedge location=location \wedge stockcnt \geq cnt} stock)) \bowtie price)$$

Part 3 Index Structures (Total: 20 Points)

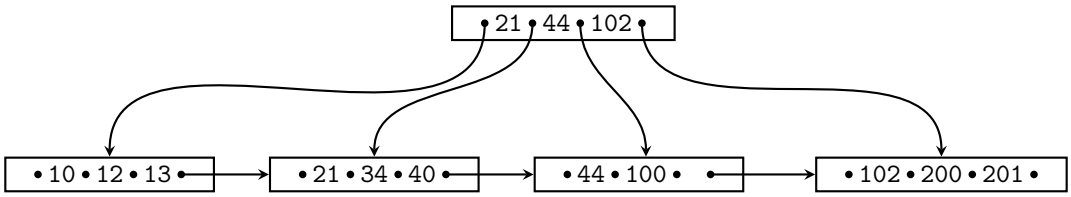
Question 3.1 B+-tree Operations (20 Points)

Given is the B+-tree shown below ($n = 3$). Execute the following operations and write down the resulting B+-tree after each step:

insert(66), insert(202), insert(70), delete(10), insert(3)

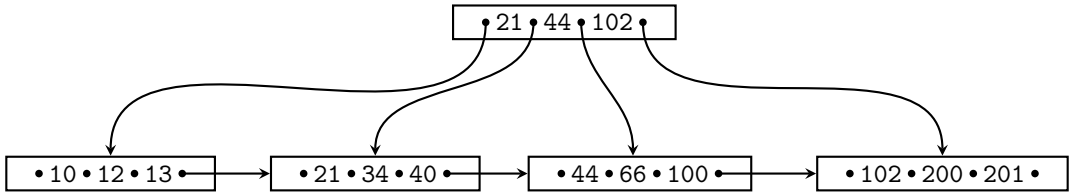
When splitting or merging nodes follow these conventions:

- **Leaf Split**: In case a leaf node needs to be split, the left node should get the extra key if the keys cannot be split evenly.
- **Non-Leaf Split**: In case a non-leaf node is split evenly, the "middle" value should be taken from the right node.
- **Node Underflow**: In case of a node underflow you should first try to redistribute and only if this fails merge. Both approaches should prefer the left sibling.

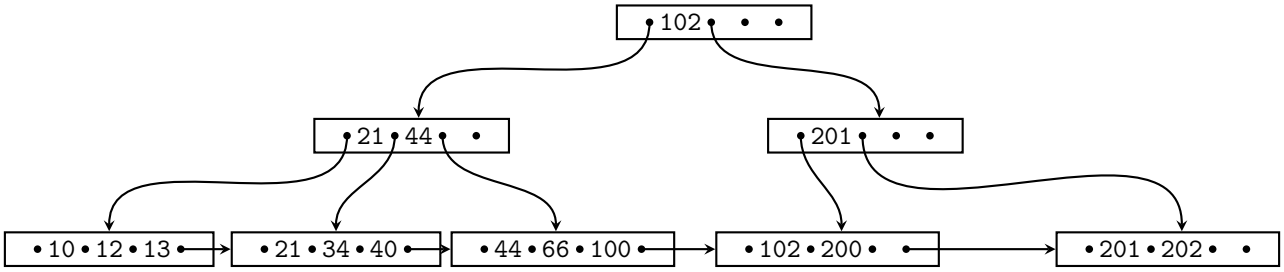


Solution

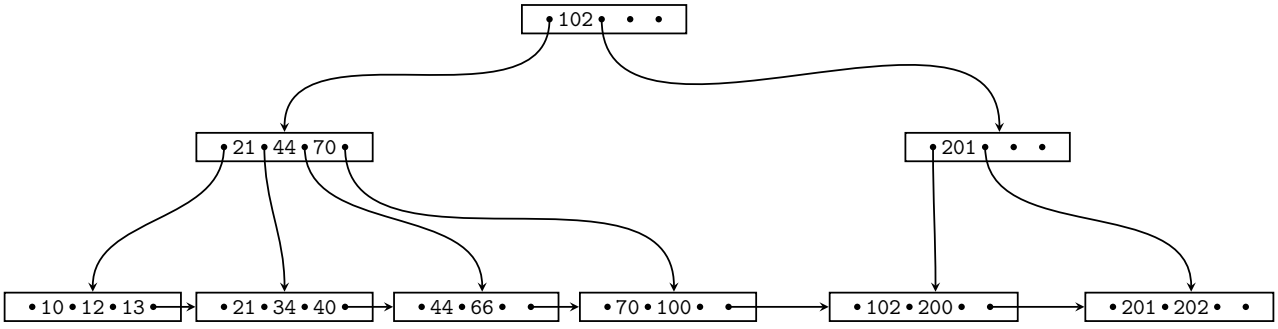
insert(66)



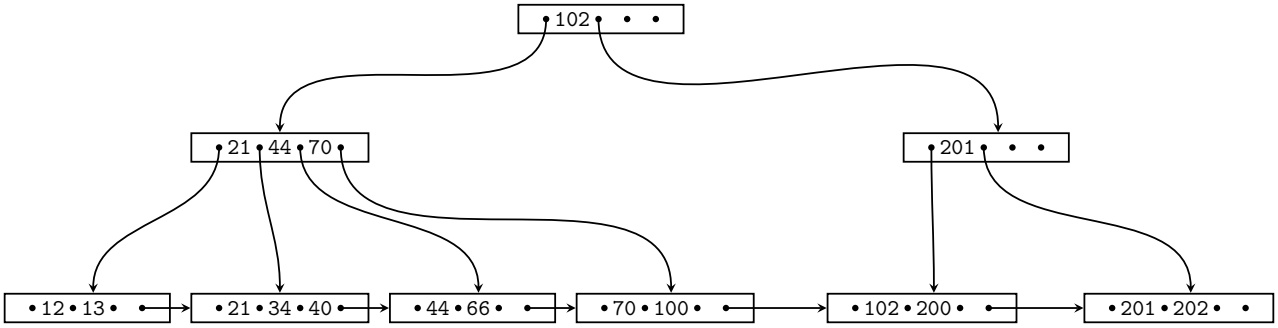
insert(202)



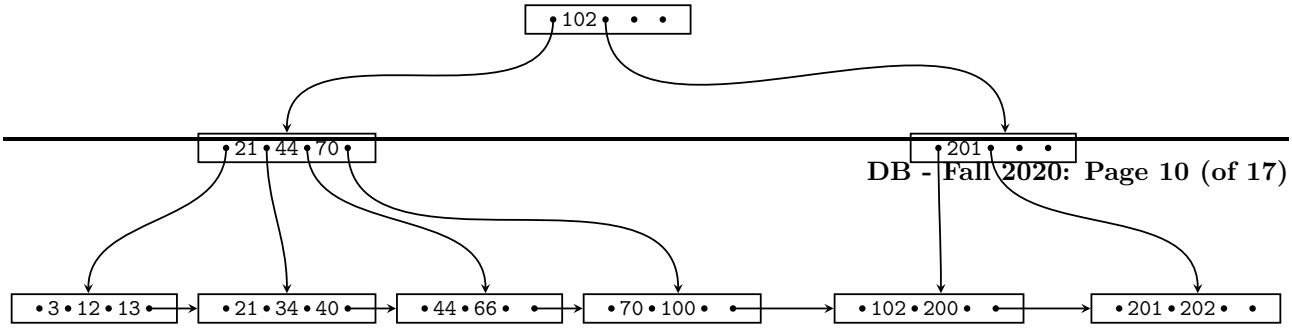
insert(70)



delete(10)



insert(3)



Part 4 I/O Cost Estimation (Total: 12 Points)

Question 4.1 External Sorting (2 Points)

You have $M = 21$ memory pages available and should sort a relation R with $B(R) = 15,000,000$ blocks. Estimate the number of I/Os necessary to sort R using the external merge sort algorithm introduced in class.

Solution

$$\begin{aligned} IO &= 2 \cdot B(R) \cdot \left(1 + \left\lceil log_{M-1}\left(\frac{B(R)}{M}\right)\right\rceil\right) \\ &= 2 \cdot 15,000,000 \cdot (1+5) \\ &= 180,000,000 \end{aligned}$$

Question 4.2 External Sorting (2 Points)

You have $M = 4001$ memory pages available and should sort a relation R with $B(R) = 10,000,000,000$ blocks. Estimate the number of I/Os necessary to sort R using the external merge sort algorithm introduced in class.

Solution

$$\begin{aligned} IO &= 2 \cdot B(R) \cdot \left(1 + \left\lceil log_{M-1}\left(\frac{B(R)}{M}\right)\right\rceil\right) \\ &= 2 \cdot 10,000,000,000 \cdot (1+2) \\ &= 60,000,000,000 \end{aligned}$$

Question 4.3 I/O Cost Estimation (8 = 2 + 3 + 3 Points)

Consider two relations R and S with $B(R) = 300,000$ and $B(S) = 200,000$. You have $M = 1,001$ memory pages available. Compute the minimum number of I/O operations needed to join these two relations using **block-nested-loop join**, **merge-join** (the inputs are not sorted), and **hash-join**. You can assume that the hash function evenly distributes keys across buckets. Justify you result by showing the I/O cost estimation for each join method.

Solution

- **BNL**: S is the smaller relation.
  $\lceil \frac{B(S)}{M-1} \rceil \cdot [B(R) + min(B(S), (M-1))] = 200 \cdot [300000 + 1000] = 60,200,000$ I/Os
- **MJ**: We can generate sorted runs of size $1,001$ that means we need 1 merge pass(es) for R and 1 merge passes for S. The number of runs in the last phase of sorting is 300 for R and 300 fro S. The optimization is applicable, because $300+200 < M$. Thus, the total cost is $3*B(R)+3*B(S) = 3*300,000+3*200,000 = 1,500,000$ I/Os.
- **HJ**: After 1 partition phase the size of the partitions for S (200 pages) is small enough to fit one partition into memory, build an in-memory hash table of each partition of S, and stream a partition of R probing the hash table. $(2 \cdot 1 + 1) \cdot (B(R) + B(S)) = 3 \cdot (300,000 + 200,000) = 1,500,000$ I/Os.

Part 5 Result Size Estimations (Total: 12 Points)

Consider the relations `order` and `stock` from the SQL part. Recall that `oid` and `(location, item)` are the primary keys of these relations.

Given are the following statistics:

$$
\begin{aligned}
T(order) &= 400,000 \\
V(order, oid) &= 400,000 \\
V(order, location) &= 30 \\
V(order, item) &= 200 \\
V(order, cnt) &= 70
\end{aligned}
\qquad min(cnt) = 1 \qquad max(cnt) = 81
$$

$$
\begin{aligned}
T(stock) &= 5500 \\
V(stock, location) &= 30 \\
V(stock, item) &= 250 \\
V(stock, stockcnt) &= 560
\end{aligned}
\qquad min(stockcnt) = 1 \qquad max(stockcnt) = 1,000
$$

Question 5.1 Estimate Result Size (4 Points)

Estimate the number of result tuples for the query $q = \sigma_{location=Chicago \vee item=Shovel}(order)$ using the first assumption presented in class (values used in queries are uniformly distributed within the active domain).

Solution

Calculate probability that a tuple fulfills the conditions using the independence assumption. For disjunction, we use the logical equivalence $a \vee b = \neg(\neg a \wedge \neg b)$

$$P(location = Chicago \vee item = Shovel) = (1 - (1 - \frac{1}{30}) \cdot (1 - \frac{1}{200}) \approx 0.0382$$

$$T(q) = T(order) * P(location = Chicago \vee item = Shovel) \approx 400,000 \cdot 0.0382 \approx 15,266.6667$$

Question 5.2 Estimate Result Size (4 Points)

Estimate the number of result tuples for the query $q = \sigma_{cnt \geq 10 \wedge cnt \leq 20}(order)$ using the first assumption presented in class.

Solution

Since the two conditions both restrict the domain of attribute `cnt` $P(cnt \geq 10 \wedge cnt \leq 20)$.

$$T(q) = \frac{20 - 10 + 1}{max(cnt) - min(cnt) + 1} \cdot T(order) = \frac{11}{81} \cdot 400,000 \approx 54320.9877$$

Question 5.3 Estimate Result Size (4 Points)

Estimate the number of result tuples for the query $q = order \bowtie_{item=item} stock$ using the first assumption presented in class.

Solution

$$T(q) = \frac{T(order) \cdot T(stock)}{max(V(order, item), V(stock, item)} = \frac{400,000 \cdot 5500}{max(200, 250)} = 8,800,000$$

Part 6 Schedules (Total: 15 Points)

Question 6.1 Schedule Classes (15 = 5 + 5 + 5 Points)

Indicate which of the following schedules belong to which class. **Every correct answer is worth 1 point. Every incorrect answer results in 1 point being deducted. You are allowed to skip questions (0 points).** Recall transaction operations are modelled as follows:

$w_1(A)$	transaction 1 wrote item A
$r_1(A)$	transaction 1 read item A
c_1	transaction 1 commits
a_1	transaction 1 aborts

$$S_1 = r_1(a), w_3(b), w_3(a), r_4(d), w_2(a), r_4(b), r_3(c), w_2(c), w_2(d), c_1, c_2, c_3, c_4$$
$$S_2 = w_1(b), w_3(c), r_4(d), w_3(d), c_3, r_2(a), w_2(b), c_2, w_4(a), c_4, r_1(c), c_1$$
$$S_3 = w_2(a), c_2, w_3(a), w_1(a), c_1, c_3$$

S_1 is recoverable	❑ no	■ yes
S_1 is cascade-less	❑ no	■ yes
S_1 is strict	■ no	❑ yes
S_1 is conflict-serializable	❑ no	■ yes
S_1 is 2PL	❑ no	■ yes

S_2 is recoverable	❑ no	■ yes
S_2 is cascade-less	❑ no	■ yes
S_2 is strict	■ no	❑ yes
S_2 is conflict-serializable	■ no	❑ yes
S_2 is 2PL	■ no	❑ yes

S_3 is recoverable	❑ no	■ yes
S_3 is cascade-less	❑ no	■ yes
S_3 is strict	■ no	❑ yes
S_3 is conflict-serializable	❑ no	■ yes
S_3 is 2PL	❑ no	■ yes