Assignment 2 **Alan Ou**
301320642

Question 1

I)

a) There is a match. Primary conjunct is: Students.ID < 2345000

b) There is a match. Primary conjunct is: Students.ID = 2345000

II)

a) Not a match since hash based indexing does not support range searches.

b) There is a match. Primary conjunct is: Students.ID = 2345000

III)

a) There is a match. Primary conjunct is: Students.ID=2345000

b) Not a match since search key < Students.ID , Students.Age > sorts relation by Students.ID, therefore if Students.Age = 22 is wanted, the entire relation would have to be searched.

c) There is a match. Primary conjuncts are: Students.ID<2345000 and Students.ID<2345000^Students.Age=22

d) There is a match. Primary conjuncts are: Students.ID=2345000 and Students.ID=2345000^Students.Age>22

IV)

a) Not a match since search key < Students.ID , Students.Age > is not a subset of the selection condition Students.ID = 2345000.

b) Not a match since search key < Students.ID , Students.Age > is not a subset of the selection condition Students.Age=22.

c) There is a match. Primary conjunct is: Students.ID=2345000^Students.Age=22

d) Not a match since not all selection conditions are equality predicates.

Question 2

a) First pass is equal to pass 0, therefore:

$\left\lceil \frac{10000}{10} \right\rceil$ = **1000 runs**

b) $\left\lceil \log_{10-1} \left\lceil \frac{10000}{10} \right\rceil \right\rceil + 1$ = **5 passes**

c) 2 x 10000 x($\left\lceil \log_{10-1} \left\lceil \frac{10000}{10} \right\rceil \right\rceil + 1$) = **100000**

d) B − 1 ≥ $\left\lceil \frac{10000}{B} \right\rceil$ => **B = 101 buffer pages**

Assignment 2 **Alan Ou**
301320642

Question 3

a) Using PNLJ: 90000 + 90000(150 000) = 1.350009×10^{10}
Using BNLJ: 150 000 + $\left\lceil \frac{150\,000}{400-2} \right\rceil (90000)$ = 34 080 000
Using SMJ: 400 > $\sqrt{150\,000} > \sqrt{90000}$

3 (150 000 + 90000) = 720 000

Using HJ: 400 $> \sqrt{90000}$

3 (150 000 + 90000) = 720 000

We could use either sort-merge join or hash join because both of these join algorithms have a cost of 720 000.

b) Using PNLJ: 90000 + 90000(150 000) = 1.350009×10^{10}
Using BNLJ: 150 000 + $\left\lceil \frac{150\,000}{50-2} \right\rceil (90000)$ = 281 400 000
Using SMJ: 50 $\not> \sqrt{150\,000} > \sqrt{90000}$ <- NOT possible because not enough buffer pages

Using HJ: 50 $\not> \sqrt{90000}$ <- NOT possible because not enough buffer pages

Since SMJ and HJ are not possible because of the shortage of buffer pages, we choose from page nested loop join and block nested loop join. We choose block nested loop join, which has a cost of 281 400 000.

c) Using PNLJ: 90000 + 90000(150 000) = 1.350009×10^{10}
Using BNLJ: 150 000 + $\left\lceil \frac{150\,000}{310-2} \right\rceil (90000)$ = 44 070 000
Using SMJ: 310 $\not> \sqrt{150\,000} > \sqrt{90000}$

Using HJ: 310 $> \sqrt{90000}$

3 (150 000 + 90000) = 720 000

The number of buffer pages is not enough for sort merge join, but is enough for hash join, therefore we choose hast join as our join algorithm. The cost for using hash join is 720 000.

Question 4

Using BNLJ: 400 + $\left\lceil \frac{400}{402-2} \right\rceil (2000)$ = 2400

We could use the same block nested loop join algorithm because using the extra 201 buffer pages, we can get the cost down to 2400.

Assignment 2

Alan Ou
301320642

Question 5

Assuming that there is enough memory to bring in all pages during pass 0, we sort all the records during pass 0 according to department, then "write them back as one sorted run." At pass 1 or also known as second pass, the file is completely sorted.

Now, after the file is sorted, to evaluate the count operation, the relation (in this case, Students) is scanned in sorted order where the aggregate operation (in this case, the count operation) is computed. The last step is to output every department name entry and the tuple count.

Question 6

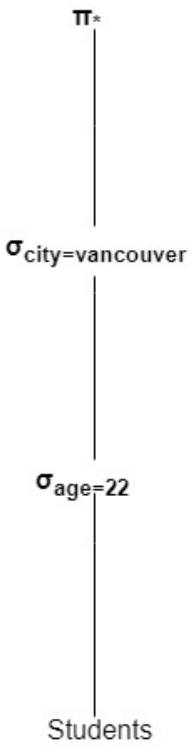
Using SMJ: $100 \not> \sqrt{50\,000} > \sqrt{10\,000}$

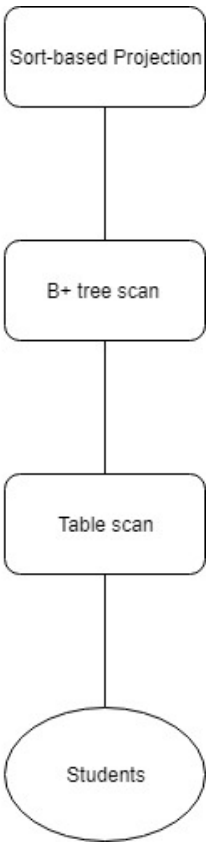
Using HJ: $100 > \sqrt{10\,000}$

3 (50 000 + 10 000) = 180 000

Although the buffer requirement is an approximation, we are still not able to sort-merge join because we the 100 buffers is far too less than the required, therefore, we will choose the algorithm that approximately satisfies the inequality, we will use the hash join algorithm. The cost of is 180 000.

Question 7





a) Provided the optimized logical query plan on the left, the physical query plan with the lowest estimated cost is as follows on the right of the logical query plan.

Assuming that we have enough buffers for sort-based project, sort-based projection would be preferred over hash based projection because results are in sorted order.

We use a B+ tree scan to select tuples that have city=Vancouver because a clustered B+ tree already exists.

We use table scan to select tuples that have age=22 because since the index is unclustered, there is a chance that each tuple will be on a different page, therefore using table scanning would be less costly.

Cost = 3(400 pages) + (2 + 500 * 0.1 + 500 * 0.1 * 0.2) + 400 = **1710**

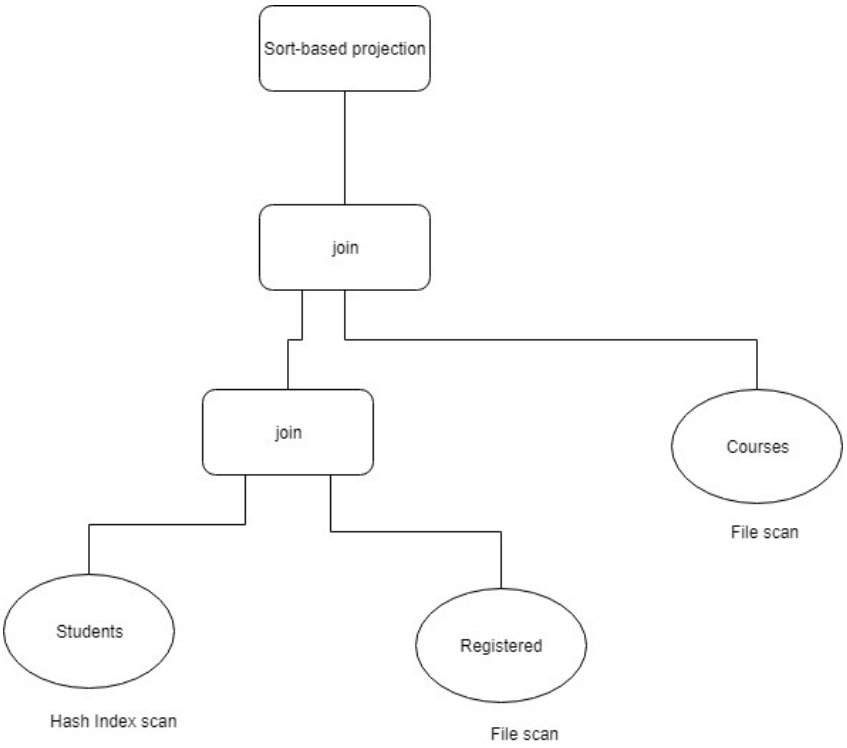
b) 36000 * 500 * 8000 = **1.44×10^{11} tuples**

c) During the first pass, the best 1-relation plans are determined for Students, Courses, and Registered relations. This is where all available access paths including file scan and index scanning are considered and chosen according to least cost.

Students relation: Hash index matches the students relation, therefore will likely be the cheapest.
Courses relation: No index, therefore file scan will likely be cheapest.
Registered relation: No index, therefore file scan will likely be cheapest.

During pass 2, the best way to join the results of each 1-relation plan to another relation is considered. During pass 3, the best way to join the result of pass 2 to the other 1-relation is found.

d)



A sort based projection is chosen because 80 > $\sqrt{1000}$ and also leaves results sorted.

File scan is chosen for courses and registered relations because there are no indexes on those relations; any other type of scanning will likely be more costly.

A hash index matches on the student relation, therefore index scanning will be best for the students relation.

Assignment 2 **Alan Ou**
301320642

Question 8

Each employee record is – 100 bytes

Each index data entry is – 20 bytes

# of pages in Employee relation – 10 000 pages

# of relations per data page – 20 relations

Height of tree – 2

a) I) sal > 100

Cost of unclustered B+ tree = 2 + (10 000 * 0.2 * 0.1) + (10,000 pages * 20 tuples/page * 0.1) =
= 20 202
Cost of using file scan = number of pages in the relation = 10 000

Therefore, the most selective access path is **file scan**.

II) age = 25

Cost of hash index = (1.2 * 20000) + (20000) = 24 000

Cost of clustered B+ tree index = 2 + 10 000 * 0.1 + 10 000 * 0.1 * 0.2 = 1 202

Cost of using file scan = number of pages in the relation = 10 000

Therefore, the most selective access path is **clustered B+ tree index**.

iii) age > 20

Cost of hash index = (1.2 * 20000) + (20000) = 24 000

Cost of clustered B+ tree index = 2 + 10 000 * 0.1 + 10 000 * 0.1 * 0.2 = 1 202

Cost of using file scan = number of pages in the relation = 10 000

Therefore, the most selective access path is **clustered B+ tree index**.

iv) eid = 1000

Cost of hash index = (1.2 * 1) + (1) = 2

The most selective access path would undoubtedly be the hash index because a hash index already exists on eid, therefore would bring the cost of retrieving the matching tuples to 1 because hash indexes have direct access (O(1)).

v) sal>200^age>30

Cost of clustered B+ tree index = 2 + 10 000 * 0.1 * 0.1 + 10 000 * 0.1 * 0.1 * 0.4 = 142

Since a clustered tree index exists on <age, sal>, an obvious choice would be to choose the clustered B+ tree index as the most selective path.

vi) sal>200^age = 20

Cost of clustered B+ tree index = 2 + 10 000*0.1*0.1 + 10 000*0.4*0.1*0.1 = 142

Since a clustered tree index exists on <age, sal>, an obvious choice would be to choose the clustered B+ tree index as the most selective path.

vii) sal>200^title = 'CFO'

Cost of unclustered B+ tree = 2 + (10 000 * 0.2 * 0.1) + (10,000 pages * 20 tuples/page * 0.1) =
= 20 202
Cost of using file scan = number of pages in the relation = 10 000

Therefore, the most selective access path is **file scan**.

viii) sal>200^age>30^title = `CFO'

Cost of clustered B+ tree index = 2 + 10 000*0.1*0.1 + 10 000*0.4*0.1*0.1 = 142

Since a clustered tree index exists on <age, sal>, an obvious choice would be to choose the clustered B+ tree index as the most selective path.

b) I) sal > 100

Cost of unclustered B+ tree = 2 + (10 000 * 0.2 * 0.1) = 202
Cost of using file scan = number of pages in the relation = 10 000

Therefore, the least expensive evaluation method is **unclustered B+ tree**.

II) age = 25

Cost of hash index = (1.2 * 20000) + (20000) = 24 000

Cost of clustered B+ tree index = 2 + 10 000 * 0.1 * 0.4 = 402

Cost of using file scan = number of pages in the relation = 10 000

Therefore, the least expensive evaluation method is **clustered B+ tree index**.

iii) age > 20

Cost of hash index = (1.2 * 20000) + (20000) = 24 000

Cost of clustered B+ tree index = 2 + 10 000 * 0.1 * 0.4 = 402

Cost of using file scan = number of pages in the relation = 10 000

Therefore, the least expensive evaluation method is **clustered B+ tree index**.

iv) eid = 1000

Cost of hash index = (1.2 * 1) + (1) = 2

The least expensive evaluation method would undoubtedly be the hash index because a hash index already exists on eid, therefore would bring the cost of retrieving the matching tuples to 1 because hash indexes have direct access (O(1)).

v) sal>200^age>30

Cost of clustered B+ tree index = 2 + 10 000 * 0.1 * 0.1 * 0.4 = 402

Since a clustered tree index exists on <age, sal>, an obvious choice would be to choose the clustered B+ tree index as the least expensive evaluation method.

vi) sal>200^age = 20

Cost of clustered B+ tree index = 2 + 10 000 * 0.1 * 0.1 * 0.4 = 402

Since a clustered tree index exists on <age, sal>, an obvious choice would be to choose the clustered B+ tree index as the least expensive evaluation method.

vii) sal>200^title = 'CFO'

Cost of using file scan = number of pages in the relation = 10 000

Therefore, the least expensive evaluation method is **file scan**.

viii) sal>200^age>30^title = `CFO'

Cost of clustered B+ tree index = 2 + 10 000 * 0.1 * 0.1 * 0.4 = 402

Since a clustered tree index exists on <age, sal>, an obvious choice would be to choose the clustered B+ tree index as the least expensive evaluation method.

c) I) sal > 100

Cost of clustered B+ tree on <age, sal> = 2 + (10 000 * 0.4) = 4002
Cost of using file scan = number of pages in the relation = 10 000

Therefore, the least expensive evaluation method is **clustered B+ tree**.

II) age = 25

Cost of hash index = (1.2 * 20000) + (20000) = 24 000

Cost of clustered B+ tree on <age, sal> = 2 + (10 000 * 0.4 * 0.1) = 402

Cost of using file scan = number of pages in the relation = 10 000

Therefore, the least expensive evaluation method is **clustered B+ tree**.

iii) age > 20

Cost of hash index = (1.2 * 20000) + (20000) = 24 000

Cost of unclustered B+ tree on <age, sal> = 2 + (10 000 * 0.4 * 0.1) = 402

Cost of using file scan = number of pages in the relation = 10 000

Therefore, the least expensive evaluation method is **unclustered B+ tree**.

iv) eid = 1000

Cost of hash index = (1.2 * 1) + (1) = 2

The least expensive evaluation method would undoubtedly be the hash index because a hash index already exists on eid, therefore would bring the cost of retrieving the matching tuples to 1 because hash indexes have direct access (O(1)).

v) sal>200^age>30

Cost of clustered B+ tree on <age, sal> = 2 + (10 000 * 0.4 * 0.1 * 0.1) = 42

Cost of using file scan = number of pages in the relation = 10 000

Therefore, the least expensive evaluation method is **clustered B+ tree**.

vi) sal>200^age = 20

Cost of clustered B+ tree on <age, sal> = 2 + (10 000 * 0.4 * 0.1 * 0.1) = 42

Cost of using file scan = number of pages in the relation = 10 000

Therefore, the least expensive evaluation method is **clustered B+ tree**.

vii) sal>200^title = 'CFO'

Cost of clustered B+ tree index on <age, sal> = 2 + (10 000 * 0.4) + (10,000 *20 * 0.1) = 24002

Therefore, the least expensive evaluation method is **file scan**.

viii) sal>200^age>30^title = `CFO'

Cost of clustered B+ tree index on <age, sal> = 2 + (10 000 * 0.4 * 0.1 * 0.1) + (10,000 *20 * 0.1) = 20042

Since a clustered tree index exists on <age, sal>, an obvious choice would be to choose the clustered B+ tree index as the least expensive evaluation method.

d) **iv)** eid = 1000

Cost of hash index = (1.2 * 1) + (1) = 2

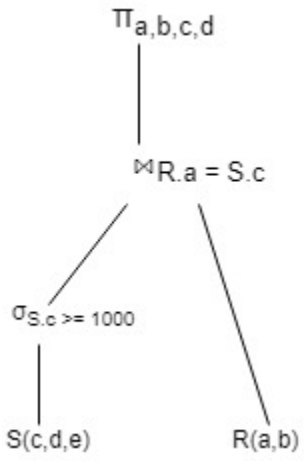
The least expensive evaluation method would undoubtedly be the hash index because a hash index already exists on eid, therefore would bring the cost of retrieving the matching tuples to 1 because hash indexes have direct access (O(1)).

e) i) File scan would be best because the best evaluation method for sal > 200 is file scan, therefore even if we choose B+ tree clustered index scanning, the cost would be more than file scanning.

ii) File scan would be best because the best evaluation method for sal > 200 is file scan and no index exists on title.

iii) Since no index occurs on title or ename, file scan would work best.

Question 9



This logical query plan has been chosen based on optimality. The cartesian product between relations S and R is replaced with a join because joins produce less results. The selection criteria $\sigma_{S.c >= 1000}$ is pushed "down the query tree for the earliest possible execution.

Extra Credit Question

According to an article by Techopedia [1], the rise of NoSQL databases was due to the big data explosion. As big data only has semi-structured, unstructured and other forms of data, a simpler, non-schema fixed structure was required. Furthermore, with the large amount of data distributed over the world, there was a need for rapid scalability and better performance. These are reasons why NoSQL databases have become more popular.

The four main types of NoSQL database management system models are:

Key-Value-Based – the most basic NoSQL model. The value of the data is stored together with a matching key without any type of structure or relation.

Column-Based – stores related data in column families

Document-Based – same as key-value, where the value of the data is the document.

Graph-Based – entities are stored with their relationships, like a tree structure with nodes connected based on relationships.

Sources

[1] Pal, K. (2016, June 07). Why the World Is Moving Toward NoSQL Databases. Retrieved July 12, 2018, from https://www.techopedia.com/2/32000/trends/big-data/why-the-world-is-moving-toward-nosql-databases