IS 19

Q3.

(a) (i)

$\Pi_{AName}(\sigma_{Gender = "Female" \wedge profit > 1,000,000}(ACTOR \bowtie ROLE \bowtie Movie))$

Or

$\Pi_{AName}(\sigma_{Gender="Female"}(ACTOR) \bowtie ROLE \bowtie \sigma_{profit>1,000,000}(MOVIE))$

(ii)

$r_1 \leftarrow Actor \bowtie ROLE \bowtie Movie$

$r_2 \leftarrow \sigma_{Gender="male" \wedge 2009 \le year \le 2019}(r_1)$

$AID\,\mathcal{G}_{sum(pay)}(r_2)$, count(MID)

(b) (i)

```
Select distinct A.AName
from ACTOR as A, ROLE as R, MOVIE as M,
where A.AID = R.AID and
      A.Gender = "Female" and
      R.MID = M.MID and
      M.Profit > 1,000,000
```

(ii)

```
Select A.AID, sum(R.Pay), count(M.MID)
from ACTOR as A, ROLE as R, MOVIE as M
where A.AID = R.AID and
      R.MID = M.MID and
      A.Gender = "male" and
      2009 ≤ M.Year ≤ 2019
Group by A.AID
```

or ((ACTOR natural inner join ROLE) natural inner join MOVIE)

(c) ACTOR: 50 Blocks
ROLE: 1000 Blocks

(i) M = 4

1) ACTOR as outer.

2 blocks at a time.

∴ 25 times in total.

read each block of R

∴ 25 × 1000 + 50

= 25050 / # of seek: $2 \cdot \frac{50}{42} = 50$

2) ROLE as outer.

1000 ÷ 2 × 50 + 1000

= 26000 / # of seek: $2 \cdot \frac{1000}{42} = 1000$

∴ choose ACTOR.

(ii) 60 - 2 > 50

∴ only 1 time needed.

∴ total transfer number is

1000 + 50 = 1050 / # of seek : 2

[if ROLE As outer :

1000 ÷ 58 = 18 -

50 × 18 + 1000

= 1900

IS 18.

Q3

(a) R: 18000 records × 10 bytes/record (attribute ×3 attributes)
÷ 3000 bytes/page.
= 60 pages = 180

S: 48000 × 10 ÷ 3000 ×3
= 160 pages = 480.

(b) n=20 — not sure 20/10?
不会! longest path: $\log_{20/2}(18000) = 5$ (4.25)
∴ At most access 5 nodes.
(usually one node = one block [ref: slides chapt. 12])
∴ The number is 5 (没用到"10 records as the answer")
根据pointer找到record. 5 + 10 = 15
最好不clustered index. second query 可以record存在不同地方

(c) R: ~~60~~ 180 pages, ~~10~~ 30 pages/partition
不会 S: ~~160~~ 480 pages, ~~27~~ 80 pages/partition

IO cost of:
Step 1: 3×60×2 = 120 (360) 把每条数据都读入Main Memory, 再写到6个不同页, 每个key的页满了之后写出 (仍在Main Memo的某一页 (butter))
Step 2: 3×160×2 = 320 (960)
Step 3: 10×2 ×6 = 1620 (180 + 480 = 660) 每个hash对 10→2 了重复6次
total: ~~2060~~ 80

(d) 不会
① 用3个page存一个partition.
在step1不必写出 (还余26 page的buffer)
② 在step2, 第一个partition可以比较完直接output, 不必储存 (属于第一pair的entri)
其余p与(b)一样.
∴ total: 2060 − 10×2 − 2×2 = 1986.
step 1: 180 + 150 (In + output)
step 2: 480 + 400 (In + 第一个P不write)
step 3: 150 + 400 (In)

hash join of R, S
① 把R中tuple依次读入, 再放入对应 partition butter, 满了后写出
② Repeat step 1 to S
③ R的一个 partition Pi 可以放入内存, 读入与S, 检查Pi每个

IS17

Q3

(a) i) CUSTOMER ⋈ RENTAL

20,000 tuples.

ii) VEHICLE ⋈ CUSTOMER

0 tuples.

(b) CUSTOMER : 50 blocks

? RENTAL : 400 blocks

i) CUSTOMER as outer:

$50 \div 2 = 25$

$50 + 25 \times 400 = 10050$

ii) RENTAT as outer:

$400 \div 2 = 200$

$400 + 200 \times 50 = 10400$

∴ CUSTOMER AS outer.

(c) For each tuple in CUSTOMER, it need to access

3 nodes(blocks)

∴ 50 (read customers into memory)

+ 5000 × (3 + 1) ← 到3 leaf 之后根据 pointer 再找 data

= 15050 20050, not sure

IS 16

Q4

(a)(1)

$r_1 \leftarrow \sigma_{c.cid = p.cid}(\rho_c(COURSE) \times \rho_p(STUDENT \bowtie ENROLL))$

$\Pi_{cname}(COURSE) - \Pi_{cname}(r_1)$

(2) ~~Π_{sname}~~ $(STUDENT \bowtie ENROLL \bowtie \sigma_{pid="p123"}(TEACH))$

P123教的所有课

$r_1 = \Pi_{cid}(\sigma_{pid="p123"} TEACH)$

$r_2 = \Pi_{sid,cid}(STUDENT \bowtie ENROLL) \div r_1$

$\Pi_{sname}(r_2 \bowtie STUDENT)$

不直接选 sname, sname is not a key.

(b)(1)

```
SELECT C.cname
from   COURSE as C
where  NOT UNIQUE(
    SELECT dept, C.cname   not sure
    from   STUDENT as S, ENROLL as E,
    Where  S.sid = E.sid and
           E.cid = C.cid )
```

对每个 course C. 执行 nested 内的语句

(2)

```
with ENROLL_NUM(cid, num) as
SELECT cid, count(Sid)
from   STUDENT natural inner join ENROLL
group by cid

with   max_enroll(num) as
SELECT max(num)
from   ENROLL_NUM
SELECT P.pname
from   PROFESSOR as P, TEACH as T, ENROLL_NUM as E, max_enroll as M.
where  P.pid = T.pid and
       T.cid = E.cid and
       E.num = M.num
```

(c) dept = "STEM" : 2,000 Ss.

credit = 2 : 100 Courses

grade = "A" : 5,000 tuples

∴ SC(dept, S) = 2000 (STUDENT as S)

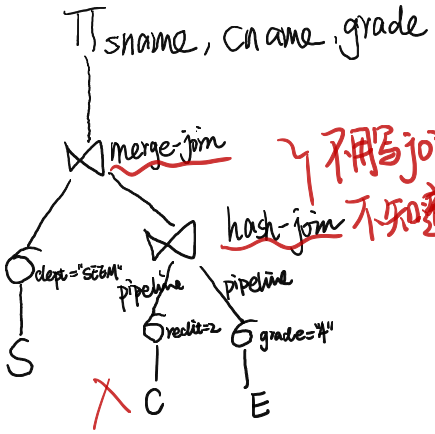
SC(credit, C) = 2 (COURSE as C)

SC(grade, E) = 5000 (ENROLL as E)

$n_S = 10{,}000$

$n_C = 200$

$n_E = 100{,}000$



瞎写的！太难了！

不用写join（属于excusion plan）不知道内存所以不能确定.

① 先做 Select 减小 table.

② 决定 join 的顺序.

i) join S and E by sid.

最多5000个.

再与 C join by cid. (5000 ⋈ 100)

5000

ii) join S and C, cartisan product

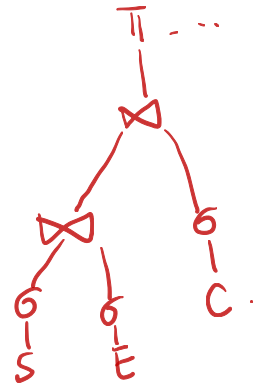
结果太大，不考虑.

iii) join C 与 E，依旧 5000.

再 join S (5000 ⋈ 2000)

依旧 5000 more expensive

∴ 选 i)



IS 15

Q4

(a) 写SQL ✓

(b) Nested-loop join ✗

(c) Compare join strategy ✗

(b). i) 600 + 1000 全读进后比对

ii) M=3. 可用1. 600 + 600×1000

M=4 2. 300×1000 + 600

(c)

i) 6000 × 2000 ,

iii) 3000