TE 30/09/2015

`Role(Actor, Movie, Character)`

Table	Tuples	Blocks
Role	400K	16K

```sql
SELECT Actor, Movie, COUNT(*) AS NumOfCharacters
FROM Role
GROUP BY Actor, Movie
HAVING COUNT(*) >= 3
```

You have different scenarios

1. `Role` is **entry sequenced** and has a secondary hash index on `Movie` with 5K buckets of 1 block each. You know there are 20K unique movies and 25K unique actors.

   - Option 1: read the entire table $16K$
   - Option 2: use the hash index, for all values of `Move` get the corresponding row and filter in memory

   $$\underbrace{5K}_{\text{Read the hash}} + \underbrace{20K}_{\text{Movies}} \cdot \underbrace{\frac{400K}{20K}}_{\text{Roles per movie}} = 405K$$

   Option 1 is the best.

2. `Role` is ordered by `Move`. No other indeces.

   We can only read the entire table: $16K$.

3. `Role` is **entry sequenced** and has a secondary B+tree on `(Actor, Movie)`, with height 3 and 3.5K leaves.

   Every leave contains the pair we need, actor and movie. Moreover, the leaves also have pointers to the data on disk. Since we are interested only in the count, we don't even have to access the `Role` data, and we can just count the pointers.

   So, reading the leaves is enough: $3.5K$.

TE 07/09/2015

`T(PK, A, B, C, RefToIDofS)`
`S(ID, X, Y)`

Table	Tuples	Blocks	Organization
T	40K	8K	Entry sequenced
S	1M	100K (buckets)	Hash on ID

Knowing that `PK` < 1000 for 2% of the tuples in `T`, that `A` is a unique attribute, and that `val(B) = 125` (homogeneously distributed), estimate the execution cost of the query below, in the following three scenarios:

1. No secondary indeces are available
2. There is a B+ index with F = 200 for `T`, on the primary key
3. There is also another B+ index on attribute `B`, with depth 3 (a root, an intermediate level, and 1.25K leaf nodes).

```sql
SELECT *
FROM S JOIN T ON RefToIDofS = ID
WHERE PK < 1000 OR (B = "pale blue" AND A <> 13472)
```

1. No secondary indeces are available

   A pure nested loop is unreasonable, as table `S` allows for effective lookups based on the ID, and the join is performed on the ID. We therefore adopt a "scan `T` and lookup in `S`" strategy, i.e., scan `T` and immediately apply the condition on the `PK` and `A`. Only for the matching tuples, perform a lookup onto `S` based on the value of their `RefToIDofS` attribute via the hash.

   We have to estimate the result size: we know that 800 tuples have `PK` < 1000, and about $\frac{40K}{125}$ for each value of `B`. The condition on `A` is irrelevant. In the worst case, we have $800 + \frac{40K}{125} = 1120$ tuples, so we'll use that.

$$\underbrace{8K}_{\text{Scan T}} + \underbrace{1.12K}_{\text{Queries on S}} = 9.12K$$

2. There is a B+ index with F = 200 for T, on the primary key

   If a B+ on `PK` is available, then the 800 tuples with a low value for `PK` are retrievable by accessing the root and the initial leaf nodes of the tree. The cost of accessing the tree is negligible in this case (it is way less than 800), so we can conclude that the cost should be about

$$\text{Tree access} + 800 \approx 800$$

   BUT we do not have any way to evaluate `B` which **is in OR**, not in AND, and hence we cannot follow this strategy: we would have to read the entire tree. It's still better to do the $9.12K$ as before.

3. There is also another B+ index on attribute B, with depth 3 (a root, an intermediate level, and 1.25K leaf nodes).

   If we also have this second B+ index, we can lookup both attributes `PK` and `B` on the respective structures. The tree on `B` has 1.25K leaf nodes for 125 colors at 10 blocks per color.

   We can read the initial leaf nodes on the B+(PK) and follow the pointers, lookup `"pale blue"` in the other B+ and follow the related pointers, eliminate the possible duplicates, and lookup onto `S`. No duplicate elimination was required previously, as each tuple of T was encountered just once in the scan, while now a tuple with low PK and of pale blue color would be extracted twice. However, duplicate elimination does not count towards disk accesses.

$$\underbrace{\sim 800}_{\text{PK<1000 with Tree}} + \underbrace{3}_{\text{B+(B) access}} + \underbrace{10}_{\text{Blocks per color in the B+(B) tree}} + \underbrace{320}_{\text{Num of pale blues}} + \underbrace{1.12K}_{\text{As in point 1}} \approx 2.2K$$