BD2 – June 20, 2017, solutions – V 0.1

Please feel free to answer this test in English, Italian, or any mixture

First Part (6 credits and 9 credits):

1. Consider a schema R(A,B,……) S(E,F,……) and the following query

```
SELECT DISTINCT   S.*
FROM              R, S
WHERE             R.A=S.E  and  R.B=0  and  S.F < 10
```

Assume that R and S are stored as heap files. The pair (A,B) is a key for R, and (E,F) is a key for S. Observe that none of the four attributes is a key – the key is the pair. Assume that unclustered RID-sorted index are defined on R.A, R.B, S.E and S.G. Assume that every index on R has 20.000 leaves, every index on S has 4.000 leaves.
Assume the following table for the optimization parameters. If you need Cardenas formula Φ(n,k), approximate it with min(n,k).
Assume that every page is 4.000 bytes long, and that every attribute uses 4 bytes.

	NReg	NPag	NLeaf of Indexes	NKey	Min	Max
R	10.000.000	200.000	20.000			
S	2.000.000	100.000	4.000			
Idx.R.A			See R	100	0	1.000
Idx.R.B			See R	100.000	0	1.000.000
Idx.S.E			See S	1.000.000	0	1.000.000.000
Idx.S.F			See S	1.000.000	0	1.000.000

a) Knowing that a pair (X,Y) is a key for a table T, and that T has 1.000.000 records, what can be said about NKey of T.X and NKey pf T.Y?

They must satisfy the same constraint that every other attribute satisfies:

$$1 \le NKey(T.X) \le NReg, \qquad 1 \le NKey(T.Y) \le NReg$$

But, moreover, they must also satisfy

$$NReg \le NKey(T.X)*NKey(T.Y)$$

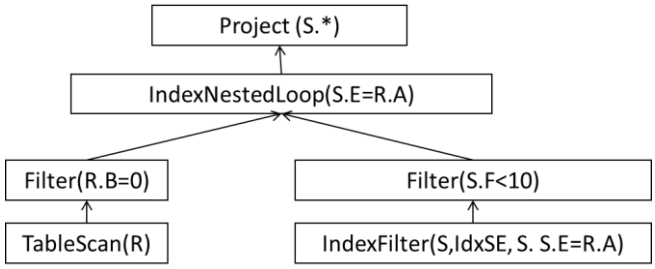
Since every element of T corresponds to a different pair (X,Y), the number of possible different pairs must be at least NReg.

b) Is DISTINCT redundant in the above query?

The closure of S.* with respect to the query condition includes R.B because of R.B=0 and and R.A because of R.A=S.E, hence it includes one key for R and one key for S, hence DISTINCT in redundant.

c) Draw an access plan that uses IndexNestedLoop, has R as its outer relation (left hand side), uses

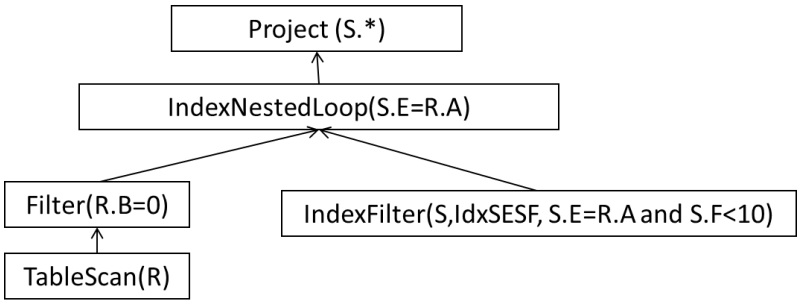
tablescan for R, and compute its cost



C(TableScan(R)) = 200.000
EReg(TableScan(R)) = 10.000.000
Fs(R.B=0) = 1/100.000
EReg(Filter(R.B=0)) = 10.000.000 / 100.000 = 100
C(IndexFilter) = CI + CD = NLeaf*fs + Φ(NReg*fs, NPag)
= 1 + Φ(2.000.000/1.000.000, 100.000) = 3
C(IndexNestedLoop) = C(O) + EReg(O)*C(I) = 200.000 + 100*3 = 200.300

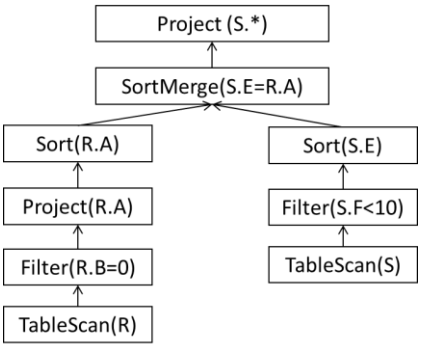
d) Could you answer to question (c) in a different way if you had a combined index on (E,F)?

With a combined index we could use the following plan:



But the cost is, in practice, very similar.

e) Draw an access plan based on MergeJoin, and compute its cost



Cost(TS(R)) = 200.000
NRec(TS(R)) = 10.000.000

NRec(Filter(R.B=0)) = 10.000.000/1.000.000 = 10
NPag(Project(R.A)) = 1
C(Sort(R.A)) = C(TS(R)) + 0 = 200.000 - the file fits one page, hence is sorted in main memory

Cost(TS(S)) = 100.000
NRec(TS(S)) = 2.000.000
NPag(Filter(S.F<10)) = 100.000*10/1.000.000 = 1
C(Sort(S.E)) = C(TS(S)) + 0 = 100.000 - the file fits one page, hence is sorted in main memory

C(Project) = C(SortMerge) = 200.000 + 100.000 = 300.000

f) When MergeJoin is used with a condition R.X=S.Y where neither attribute is a primary key, MergeJoin may be forced to go back and forth. When does that happen? May that happen in this case?

It happens when a value for R.X is repeated and the same value is repeated for S.Y. This may not happen in this case: since the pair (A,B) is a key for R, then the attribute A is a key for $\sigma_{B=0}(R)$.

2. Consider the following log content. Assume that the DB was identical to the buffer before the beginning of this log, and consider a **no-undo** / redo protocol. Beware, it is **no-undo**.

(begin,T1) (W,T1,A,1,20) (begin,T2) (W,T2,B,1,20) (begin-ckp,{T1,T2}) (W,T1,A,20,40) (end-ckp) (begin,T3) (begin,T4) (W,T3,C,1,40) (commit,T2) (W,T3,C,40,60) (W,T4,B,20,80) (commit,T4)

a. Before starting this log, what was the content of A, B and C in the PS (Persistent Store)?

It was A=1, B=1, C=1

b. What happens during the checkpoint that is reported in the log? Keeping in mind that this is no-undo, which pages are written, or may be written, to the PS?

No page is written, since T1 and T2 did not commit yet, and this is a no-undo system.

c. Assume there was a crash at the end of the logging period. At crash time, what was the content of A, B and C in the buffer? What can be said about the content of A, B and C in the PS?

In the buffer: A=40, B=80, C=60. In the PS: since only T2 and T4 did commit, the only operation that may have affected the PS are (W,T2,B,1,20) (W,T4,B,20,80). Hence, in the PS, A=1, C=1, and B may be either 1 or 20 or 80.

d. At restart time, which transactions are undone? Which are redone?

T1 and T3 are undone. T2 and T4 are redone.

e. List the operations that are redone, in the order in which are redone

(W,T2,B,1,20) (W,T4,B,20,80)

f. After restart is finished, what is the content of A, B and C in the buffer?

After restart, B is 80. A and C may not be in the buffer at all but, if they were, their values would be 1 and 1.

g. Undo and Redo are executed in the buffer or on the PS?

In the buffer

h. After restart is finished, what is the content of A, B and C in the PS?

As for question c: after restart is finished, in the PS, A=1, C=1, and B may still be either 1 or 20 or 80.

Second Part, BSA only (9 credits only):

3. Consider the following document that records data about bank accounts. For each account, it records the holders – a set of clients – and information about money transfers that originate from the account. For each money transfer it records date, money, and issuers: generally speaking, when a set of clients hold an account together, every transfer is issued by a non-empty subset of these holders.

```
Account*
  @IdA
  Bank
  Holder*
    @IdClient
  Transfer*
    @IdTransfer
    Year
    Date
    Amount
    IssuedBy*
        @IdClient
Client*
  @IdClient
  Name
  Country
```

a. Assume that $doc is bound to the above document. Write an XQuery query that, for each client, lists the information about the transfers that have been **issued** by that client (maybe with others), grouped by account and by year, according the following format

```
Client*
  Name
  Account*
    @IdA
    ByYear*
        Year
        TotalTransfersFromAccountByTheClientInYear (please abbreviate as TTFA)
```

```
  for $c in $doc/Client
  return <Client> {
        $c/Name,
        let $accounts = $doc/Account[.//IssuedBy/@IdClient = $c/@IdClient]
        for $a in $accounts
        return <Account> {
```

```
            $a/@IdA,
            for $y in fn:distinct-values($accounts//Year)
            return
               <ByYear>
                  <Year>{$y}</Year>
                  <TTFA>
                     {fn:sum($doc//Transfer[.//IssuedBy/@ IdClient =$c/@IdClient]/Amount)}
                  </TTFA>
               </ByYear>
            } </Account>
   }</Client>
```

b. Write and XQuery query that returns the IdTransfer of those transfers where at least one of the issuers does not belong to the set of the holders of the account

```
for $t in $doc//Transfer
where some $c in $t/IssuedBy/@IdClient
      satisfies not ($c = $t/../Holder/@IdClient)
return $t/@IdTransfer
```

4. Consider and RDF graph with classes Client, Account, Bank, and predicates

HasHolder $\subseteq$ Account $\times$ Client
HasBank $\subseteq$ Account $\times$ Bank
SharesAccountWith $\subseteq$ Client $\times$ Client
FinecoClient, HasFinecoOnly $\subseteq$ Client

Formalize the following statements in OWL, paying extreme attention to the direction of the implication:

a. Formalize in OWL: If two clients C1 and C2 are both holders of one account (or more than one), then C1 SharesAccountWith C2

```
SubClassOf ( ObjectPropertyChain ( ObjectInverseOf (HasHolder) HasHolder )
                SharesAccountWith
)
```

b. Define ‘reflexive’ a relation such that x R y implies that x R x and y R y. Is it possible to deduce from (a) that SharesAccountWith is reflexive?

The binary relation ObjectPropertyChain (ObjectInverseOf (HasHolder) HasHolder) is reflexive, but this does not imply that SharesAccountWith is reflexive, since (a) only gives a lower bound on the property.

c. Formalize in OWL: Every Account has exactly one Bank

```
SubClassOf (Account ObjectExactCardinality (1 HasBank) )
```

d. Formalize in OWL the following, assuming (c): a client C belongs to FinecoClient iff it is holder of at least one Account whose Bank is “Fineco”, where Fineco is one element of Bank

```
EquivalentClasses (FinecoClient
                ObjectSomeValuesFrom ( ObjectInverseOf (HasHolder)
                                        ObjectHasValue ( HasBank Fineco ) ) )
```

e. Formalize in OWL the following, assuming (c): a client C belongs to HasFinecoOnly iff every Account of which it is holder has Bank “Fineco”

```
EquivalentClasses (FinecoClient
                    ObjectAllValuesFrom ( ObjectInverseOf (HasHolder)
                                          ObjectHasValue ( HasBank Fineco ) ) )
```

Consider an arbitrary RDF graph the only contains triples whose predicate is either HasHolder or HasBank, Considering that graph and the five statements above, the fact that OWL has an open world semantics, and considering the directions of the implications, specify which of the following statements about a Client X may be proved and which may never be proved. For those that may be proved give an example of and RDF graph supporting the proof, for those that may never be proved give an extremely brief reason for that

f. X is in FinecoClient

Yes, for example with the following graph: C HasHolder john. C HasBank Fineco.

g. X is not in FinecoClient

Cannot be proved because of the Open World Assumption

h. X is in HasFinecoOnly

Cannot be proved because of the Open World Assumption

i. X is not in HasFinecoOnly

Cannot be proved since it is not possible to prove that a Bank is not Fineco

j. Would (i) change if you could add a DifferentIndividual statement?

Yes, for example from the following knowledge base we could deduce that X is not in HasFinecoOnly

C HasHolder john. C HasBank MPS. DifferentIndividual (Fineco, MPS)

k. Provide an example of a knowledge base that may allow one to prove that X is not in FinecoClient. The knowledge base may contain any OWL statement but should not explictly name the ‘FinecoClient’ property

We may use the one of j and add that X has only one account:

```
ClassAssertion ( ObjectMaxCardinality ( 1 ObjectInverseOf ( HasHolder ) )
                 john
)
```

Or we may use jack who has no account:

```
ClassAssertion ( ObjectMaxCardinality ( 0 ObjectInverseOf ( HasHolder ) )
                 jack
)
```