I. 1.

A schedule is a series of operations (reads, writes, commit, rollback) between one transaction to another, used to preserve the order of operations in each transaction.

2. Index Nested Loops Join

T1 = Table 1

T2 = Table 2

foreach tuple $t_1$ in T1 do

  foreach tuple $t_2$ in T2 do where $t_{1i} = t_{2j}$

    add the tuple $< t_1, t_2 >$ to the result

Let: M = nr of pages in T1, $p_M$ records / page

N = nr of pages in T2, $p_N$ records / page

- If the inner relation (T2 in our case) has an index on the join column, we can use it.

- cost of computation: $M + (p_M \cdot M \cdot$ cost of finding the matching tuples in T2)

- the cost for searching an index on the outer relation is usually 1.2 for a hash index and the cost of retrieving the corresponding tuples is 1 / tuple (if the index is clustered) and up to one I/O per an unclustered index.

Example: T: 800 pages, 100 records/page
R: 500 pages, 80 records/page

- if we have a hash index on TID on T
- scan R: 500 pages = 500 I, O
500 · 80 tuples
- if we assume the records of T are uniformly distributed => 80000 / 40000 = 2 records

=> Total cost: 500 + 500 · 80 · (1.2 + 1) = 88 500 I/O
for a clustered index
500 + 500 · 80 · (1.2 + 2) = 128 500 I/O
for an unclustered index

II. 1. Conflict relation:

conf(S) = { 1. ($R_1(A)$, $W_2(A)$)
2. ($R_2(A)$, $W_3(A)$)
3. ($W_2(A)$, $W_3(A)$)

In order to S being conflict serializable, the order of the operations in the schedule ($T_1$ $T_2$ $T_3$) must preserve the order of the operations in conf(S):

$T_1 T_2 T_3 =$ $R_1(A)$
$W_2(F)$
$R_1(D)$
$W_3(B)$
$R_2(A)$
$W_2(A)$
$W_1(E)$
$R_2(D)$
$W_3(A)$
$R_3(C)$
$R_3(B)$
$R_3(D)$

1st conflict

2nd conflict

3rd conflict

We can observe that the order of operations is preserved $\Rightarrow S$ is conflict serializable

$S_{ser}$ :

| T1 | T2 | T3 |
|---|---|---|
| R(A) | | |
| R(D) | | |
| W(E) | | |
| | W(F) | |
| | R(A) | |
| | W(A) | |
| | R(D) | |
| | | W(B) |
| | | W(A) |
| | | R(C) |
| | | R(B) |
| | | R(D) |

2.

$\Pi_{A.IDA, MIN(C.Value2)}$ (
HAVING COUNT(*) <= 100 (
GROUP BY A.IDA (
$\Pi_{A.IDA, C.Value2}$ (
$\sigma_{A.IDA=B.IDA \wedge B.IDC=C.IDC \wedge A.Name='Tourcu' \wedge}$

C.Value1=100 (A x B x C))))

3.

T1 : 500.000 records; 100 rec/page => 5000 pages

T2 : 100.000 records; 100 rec/page => 1000 pages

a) Block Nested Loops join:

Total Cost: 1000 + 10·5000 = 51 000 I/O

b) Sort-Merge Join:

Total Cost: 2·2·5000 + 2·2·1000 + 5000 + 1000 = 30000 I/O

II. 1. e
2. a
3. e
4. a, b
5. a
6. a, c
7. b, d
8. a, b, c, d
9. a, c

10. UVFAWMZNZAVATXNATFAIJTEHVUNKBVWAEQOSSFG