records, and they take 1.5*4907.4/138.8=53.0 blocks. So time for this query will take 20+53*10=550ms
The total time is 20+550=570

how to make the assumption is on
http://www.ccs.neu.edu/home/kenb/cs3200/resources/a5ans.html)

Secondary B-Tree
The first query will retrieve the red records, and this takes 1010ms (as in (2)).
The second query will retrieve 5000 records, and this takes 10+10*5000=50010ms
The total time=51020ms

(4) Insert a new color
There is no color table, so all time is 0.

(5) Modify the color of a paint
Heap:
There are 1M paints, and they take 3661.7 blocks, so we need on average 10+(3661.7 -1)/2*0.1=193.0ms to find the paint.
Then we modify it and write it back, which takes another 193.0ms
The total time=386ms

Primary B-Tree:
It takes 3 random I/Os.
The total time=30ms

Secondary B-Tree:
Same as Primary B-Tree
The total time=30ms

You can find this on http://www.ccs.neu.edu/home/kenb/cs3200/resources/a5ans.html

HW8

I use the following statement from the book:
Every conflict serializable schedule is serializable, if we assume that the set of items in the database does not grow or shrink. Every conflict serializable schedule is view serializable, although the converse is not true.

A schedule is said to be strict if a value written by a transaction T is not read or overwritten by other transactions until T either aborts or commits.

1. R1X, R3Z, W2Y, R3Z, R2Z, W1Y, W2Z, R1X, W3Z, C3, C1, C2
   R3(z)->W2(z),

W2(y)->W1(y)

R3(z)->W2(z)

R2(z)->W3(z)

Not Conflict serializable

6 Possible serial orders:

R1(x), W1(y), R1(x), C1, W2(y), R2(z), W2(z), C2, R3(z), R3(z), W3(z), C3

The first R3(z) in the original schedule reads the value of (z)

The first R3(z) in the serial order reads the value of W2(z)

R1(x), W1(y), R1(x), C1, R3(z), R3(z), W3(z), C3, W2(y), R2(z), W2(z), C2

R2(z) in the original schedule reads the value of (z)

R2(z) in the serial order reads the value of W3(z)

W2(y), R2(z), W2(z), C2, R3(z), R3(z), W3(z), C3, R1(x), W1(y), R1(x), C1

The first R3(z) in the original schedule reads the value of (z)

The first R3(z) in the serial order reads the value of W2(z)

W2(y), R2(z), W2(z), C2, R1(x), W1(y), R1(x), C1, R3(z), R3(z), W3(z), C3

The first R3(z) in the original schedule reads the value of (z)

The first R3(z) in the serial order reads the value of W2(z)

R3(z), R3(z), W3(z), C3, R1(x), W1(y), R1(x), C1, W2(y), R2(z), W2(z), C2

R2(z) in the original schedule reads the value of (z)

R2(z) in the serial order reads the value of W3(z)

R3(z), R3(z), W3(z), C3, W2(y), R2(z), W2(z), C2, R1(x), W1(y), R1(x), C1

R2(z) in the original schedule reads the value of (z)

R2(z) in the serial order reads the value of W3(z)

Thus not view serializable

Not seriablizable

Not strict: because W2(y) happens before W1(y), but C2 ends after C1.

2. W1(X), R1(Z), R3(X), R1(X), R2(Z), R3(Y), C1, W2(Z), W2(Y), R2(Y), W3(X), C3, C2

   W1(x)->R3(x)

   W1(x)->W3(x)

   R1(z)->W2(z)

   R1(x)->W3(x)

   R3(y)->W2(y)

   Conflict-serializable,

   Each conflict serializable schedule is view serializable, thus it's view serializable

   Every conflict serializable schedule is serializable, thus it's serializable

   Not Strict: because W1(x) happens before R3(x), but T1 commits after R3(x)

3. R2(Z), W2(Z), W1(X), W2(Y), R3(X), R1(Z), R3(Y), W3(X), R2(Y), W1(Z), C1, C2, C3

R2(z)->W1(z)
W2(z)->R1(z)
W2(z)->W1(z)
W1(x)->R3(x)
W1(x)->W3(x)
W2(y)->R3(y)
Conflict-serializable,
Each conflict serializable schedule is view serializable, thus it's view serializable
Every conflict serializable schedule is serializable, thus it's serializable
Not Strict: because W2(z) happens before R1(z) but C2 ends after C1

4. R3(X), R2(Z), W1(X), R1(Z), W2(Z), R1(X), W2(Y), R3(Y), W3(X), C3, W1(Z), C1, R2(Y), C2
R3(x)->W1(x)
R2(z)->W1(z)
W1(x)->W3(x)
R1(z)->W2(z)
W2(z)->W1(z)
R1(x)->W3(x)
W2(y)->R3(y)
Not Conflict serializable

"R3(y) reads W2(y)" means C2 comes before C3. Thus only 3 possible serial orders.
W1(x), R1(z), R1(x), W1(z), C1, R2(z), W2(z), W2(y),R2(y),C2,R3(x),R3(y),W3(x),C3
R3(x) in the original schedule reads the value of (x)
R3(x) in the serial order reads the value of W1(x)
R2(z), W2(z), W2(y),R2(y),C2,R3(x),R3(y),W3(x),C3, W1(x), R1(z), R1(x), W1(z), C1
R1(z) in the original schedule reads the value of (z)
R1(z) in the serial order reads the value of W2(z)
R2(z), W2(z), W2(y),R2(y),C2, R1(z), R1(x), W1(z), C1, R3(x),R3(y),W3(x),C3
R1(z) in the original schedule reads the value of (z)
R1(z) in the serial order reads the value of W2(z)
Not view serializable
Not serializable
Not Strict: because W1(x) happens before W3(x), but C3 ends before C1.

5. R1(A), R1(B), W1(A), R2(A), W1(B), R3(C), C1, W2(C), W2(B), R2(C), W3(A), C3, C2
R1(a)->W3(a)
R1(b)->W2(b)
W1(a)->R2(a)
W1(a)->W3(a)
R2(a)->W3(a)
W1(b)->W2(b)
R3(c)->W2(c)
Not Conflict serializable

"R2(a) reads W1(a)" means C1 comes before C2. Thus only 3 possible serial orders.

R1(a),R1(b),W1(a),W1(b),C1,R2(a),W2(c),W2(b),R2(c),C2,R3(c),W3(a),C3

R3(c) in the original schedule reads the value of (c)

R3(c) in the serial order reads the value of W2(c)

R1(a),R1(b),W1(a),W1(b), C1,R3(c),W3(a),C3, R2(a),W2(c),W2(b),R2(c),C2

R2(a) in the original schedule reads the value of W1(a)

R2(a) in the serial order reads the value of W3(a)

R3(c),W3(a),C3, R2(a),W2(c),W2(b),R2(c),C2, R1(a),R1(b),W1(a),W1(b), C1

R2(a) in the original schedule reads the value of W1(a)

R2(a) in the serial order reads the value of W3(a)

Not view serializable

Not serializable

Not Strict: because W1(a) happens before R2(a) but T1 commits after R2(a).

6. R2(A), W2(C), R1(A), W2(B), R1(B), R3(C), W3(A), R2(C), W1(B), C1, C2, C3

   R2(a)->W3(a)

   W2(c)->R3(c)

   R1(a)->W3(a)

   W2(b)->W1(b)

   Conflict-serializable,

   Each conflict serializable schedule is view serializable, thus it's view serializable

   Every conflict serializable schedule is serializable, thus it's serializable

   Not Strict: "W2(b)->W1(b)" means C2 should commit before C1, but in the schedule we have C1 commits before C2.

7. R3(B), R2(A), R3(C), W3(A), R1(A), R1(B), W2(C), C3, W1(A), W2(B), W1(B), C1, R2(C), C2

   R3(b)->W2(b)

   R3(b)->W1(b)

   R2(a)->W3(a)

   R2(a)->W1(a)

   R3(c)->W2(c)

   W3(a)->R1(a)

   W3(a)->W1(a)

   R1(b)->W2(b)

   W2(b)->W1(b)

   Not Conflict serializable

"W3(a)->R1(a)" means C3 comes before C1. Thus only 3 possible serial orders.

R3(b),R3(c),W3(a),C3,R1(a),R1(b),W1(a),W1(b),C1,R2(a),W2(c),W2(b),R2(c),C2

R2(a) in the original schedule reads the value of (a)

R2(a) in the serial order reads the value of W1(a)

R3(b),R3(c),W3(a),C3, R2(a),W2(c),W2(b),R2(c),C2, R1(a),R1(b),W1(a),W1(b),C1

R2(a) in the original schedule reads the value of (a)

R2(a) in the serial order reads the value of W3(a)

R2(a),W2(c),W2(b),R2(c),C2, R3(b),R3(c),W3(a),C3,R1(a),R1(b),W1(a),W1(b),C1

R3(b) in the original schedule reads the value of (b)

R3(b) in the serial order reads the value of W2(b)

Not view serializable

Not serializable

Not Strict: "W2(b)->W1(b)" means C2 should commit before C1, but in the schedule C1 commits before C2.

8. R1(A), R1(B), W1(A), R2(A), W1(B), R3(C), W3(A), C3, C1, W2(C), W2(B), R2(C), C2

R1(a)->W3(a)

R1(b)->W2(b)

W1(a)->R2(a)

W1(a)->W3(a)

R2(a)->W3(a)

W1(b)->W2(b)

R3(c)->W2(c)

Not Conflict-serializable,

"W1(a)->R2(a)" means C1 should come before C2, Thus only 3 possible serial orders.

R1(a),R1(b),W1(b),C1,R2(a),W2(c),W2(b),R2(c),C2,R3(c),W3(a),C3

R3(c) in the original schedule reads the value of (c)

R3(c) in the serial order reads the value of W2(c)

R1(a),R1(b),W1(b),C1, R3(c),W3(a),C3,R2(a),W2(c),W2(b),R2(c),C2

R2(a) in the original schedule reads the value of W1(a)

R2(a) in the serial order reads the value of W3(a)

R3(c),W3(a),C3, R1(a),R1(b),W1(b),C1,R2(a),W2(c),W2(b),R2(c),C2

R1(a) in the original schedule reads the value of (a)

R1(a) in the serial order reads the value of W3(a)

Not view serializable

Not serializable

Not Strict: "W1(a)->W3(a)" means C1 should commit before C3, but in the schedule C3 commits before C1.

9. R1(A), R2(A), W2(C), W2(B), R1(B), R3(B), W1(A), R2(C), C2, R3(C), W1(B), W3(A), C3, C1

R1(a)->W3(a)

R2(a)->W1(a)

R2(a)->W3(a)

W2(c)->R3(c)

W2(b)->R1(b)

W2(b)->R3(b)

W2(b)->W1(b)

R3(b)->W1(b)

W1(a)->W3(a)

Not Conflict serializable

“W2(c)->R3(c)” and “W2(b)->R1(b)” means C2 should commit before the start of C3 and C1. Thus only 2 possible serial orders.

R2(a),W2(c).W2(b),R2(c),C2,R1(a),R1(b),W1(a),W1(b),C1,R3(b),R3(c),W3(a),C3

R3(b) in the original schedule reads the value of W2(b)

R3(b) in the serial order reads the value of W1(b)

R2(a),W2(c).W2(b),R2(c),C2, R3(b),R3(c),W3(a),C3,R1(a),R1(b),W1(a),W1(b),C1

R1(b) in the original schedule reads the value of W2(b)

R1(b) in the serial order reads the value of R3(b)

Not view serializable

Not serializable

Not Strict: “W1(a)->W3(a)” means C1 should commit before C3, but in the schedule C3 commits before C1.

10. R2(A), W2(C), R1(A), R1(B), R3(B), W2(B), R2(C), C2, R3(C), W1(A), W1(B), W3(A), C1, C3

R2(a)->W1(a)

R2(a)->W3(a)

W2(c)->R3(c)

R1(a)->W3(a)

R1(b)->W2(b)

R3(b)->W2(b)

R3(b)->W1(b)

W2(b)->W1(b)

W1(a)->W3(a)

Not Conflict serializable.

“W2(c)->R3(c)” means C2 comes before C3. Thus only 3 possible serial orders.

R2(a),W2(c),W2(b),R2(c),C2,R1(a),R1(b),W1(a),W1(b),C1,R3(b),R3(c),W3(a),C3

R3(b) in the original schedule reads the value of (b)

R3(b) in the serial order reads the value of W1(b)

R2(a),W2(c),W2(b),R2(c),C2, R3(b),R3(c),W3(a),C3,R1(a),R1(b),W1(a),W1(b),C1

R1(a) in the original schedule reads the value of (a)

R1(a) in the serial order reads the value of W3(a)

R2(a),W2(c),W2(b),R2(c),C2, R1(a),R1(b),W1(a),W1(b),C1,R3(b),R3(c),W3(a),C3

R3(b) in the original schedule reads the value of (b)

R3(b) in the serial order reads the value of W1(b)

Not view serializable

Not serializable

Not Strict: W1(a) happens before W3(a) but T1 commits after W3(a)

HW9

Q1:the keys are CDA and CDE

A->B is redundant, thus we can decompose it to a BCNF

ABCDE

/\