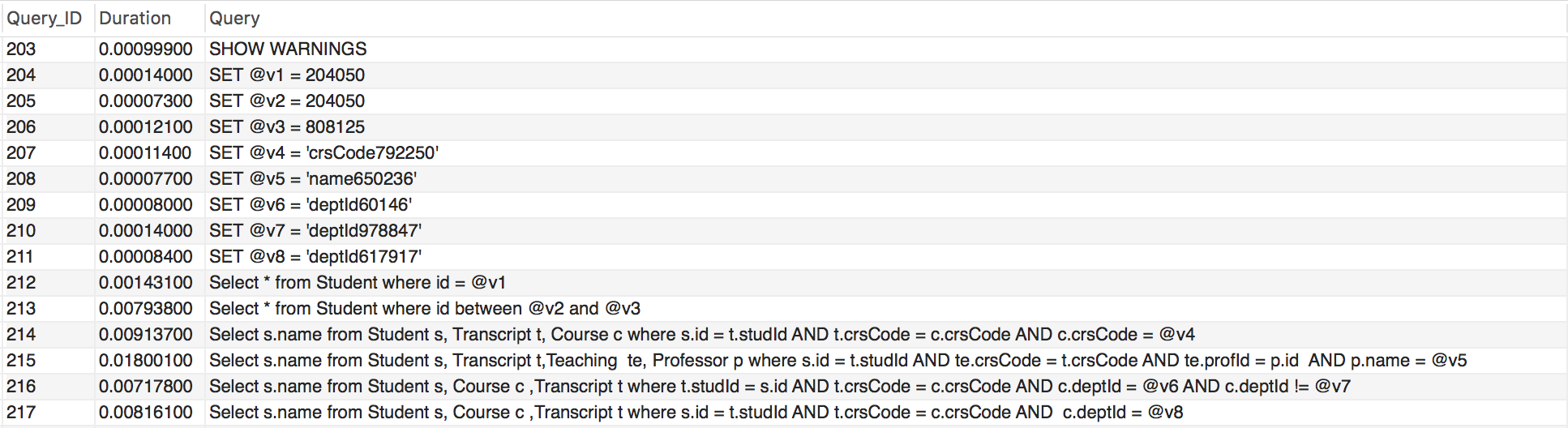
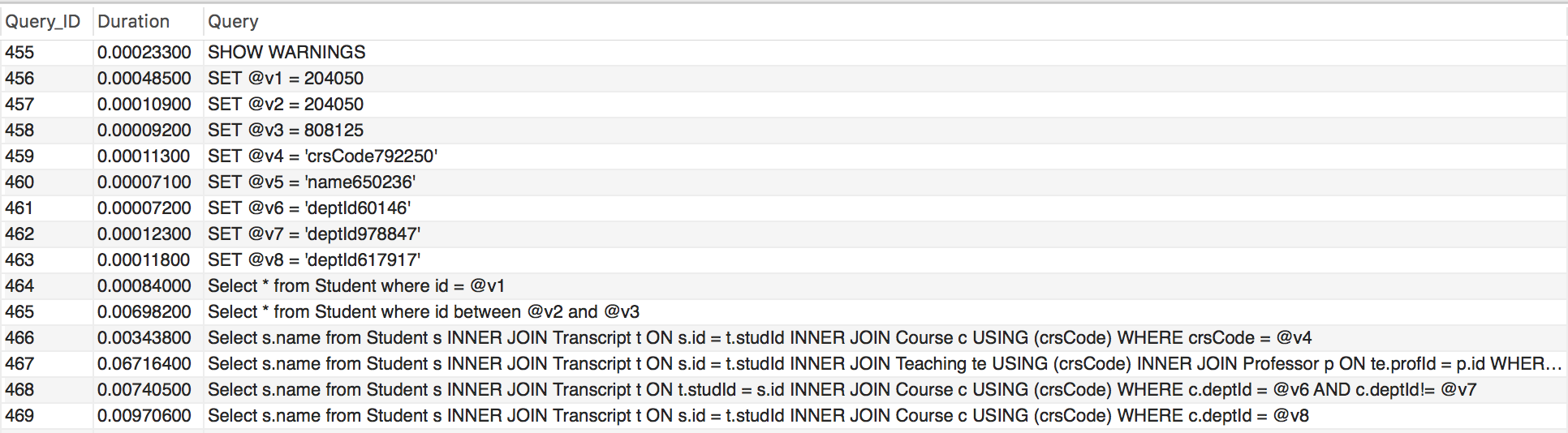
Before Optimization Query Durations

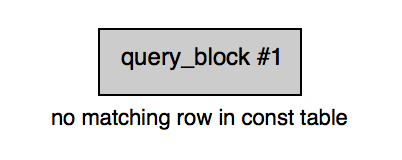


After Optimization Query Durations



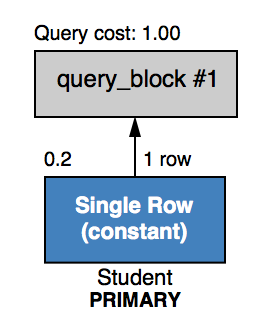
Query 1:

Before Optimization:



* Select \* from Student where id = @v1;
* Original duration = 0.00143100

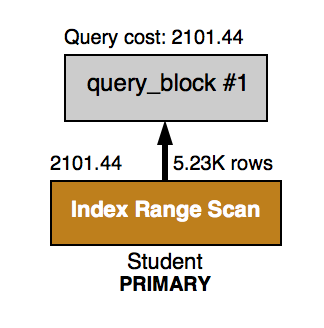
After Optimization:



* Query syntax did not change, but adjusted with index for Student names ascending called sInd.
* Optimized duration = 0.00084000

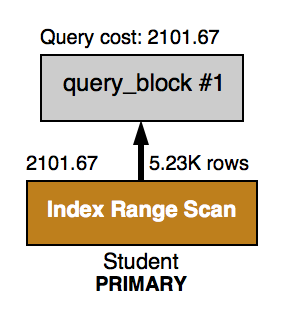
Query 2:

Before Optimization:



* Select \* from Student where id between @v2 and @v3;
* Original duration = 0.00793800

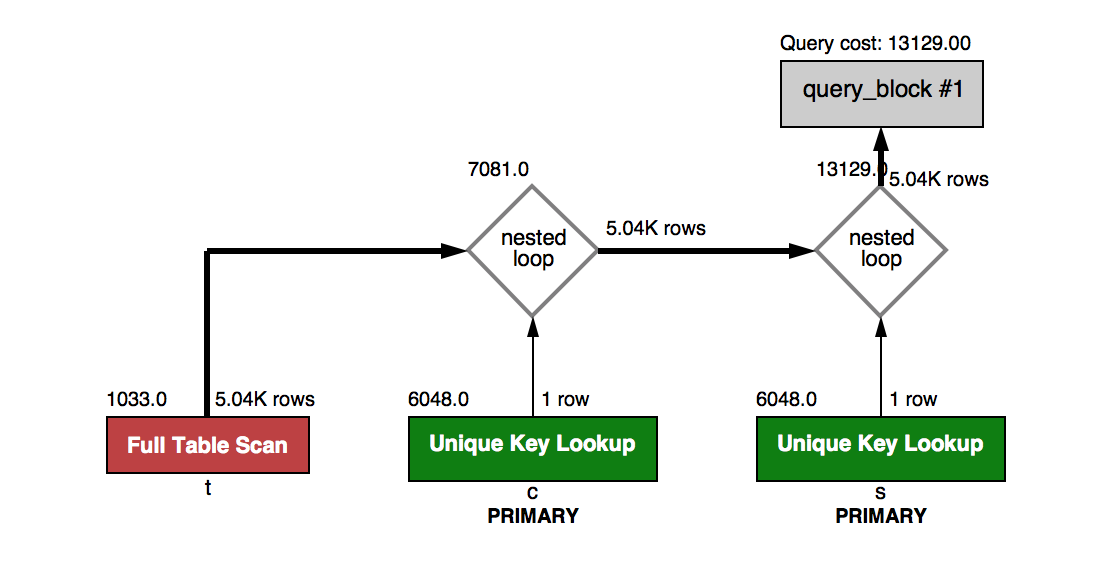
After Optimization:



* Adjusted with index for Student name ascending called sInds.
* Optimized duration: 0.00698200
* While query cost did not have a significant change, the query was performed more quickly. Range type cannot be optimized significantly.

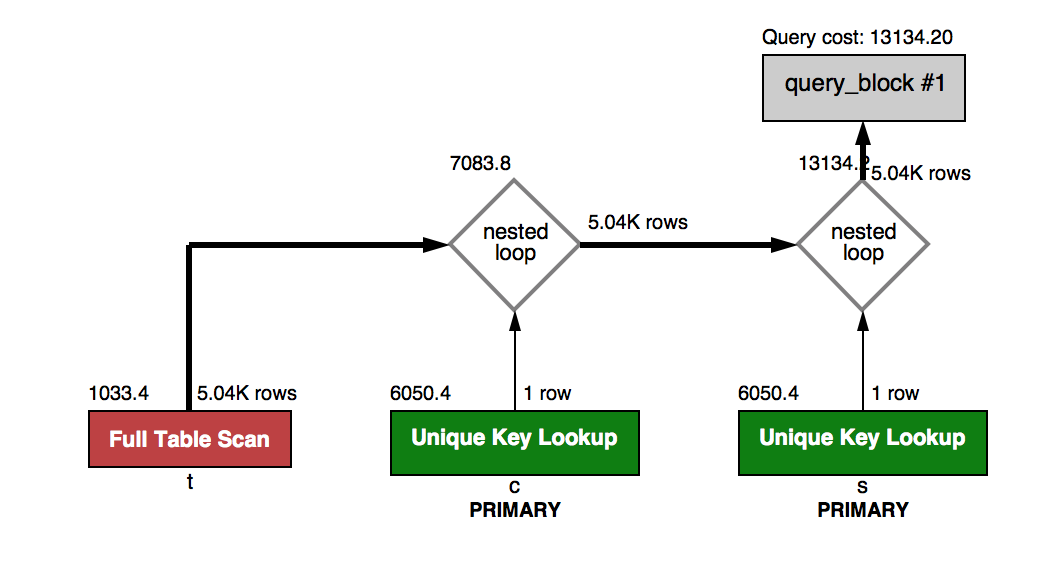
Query 3:

Before Optimization:



* Select s.name from Student s, Transcript t, Course c where s.id = t.studId AND t.crsCode = c.crsCode AND c.crsCode = @v4;
* Original duration = 0.00913700

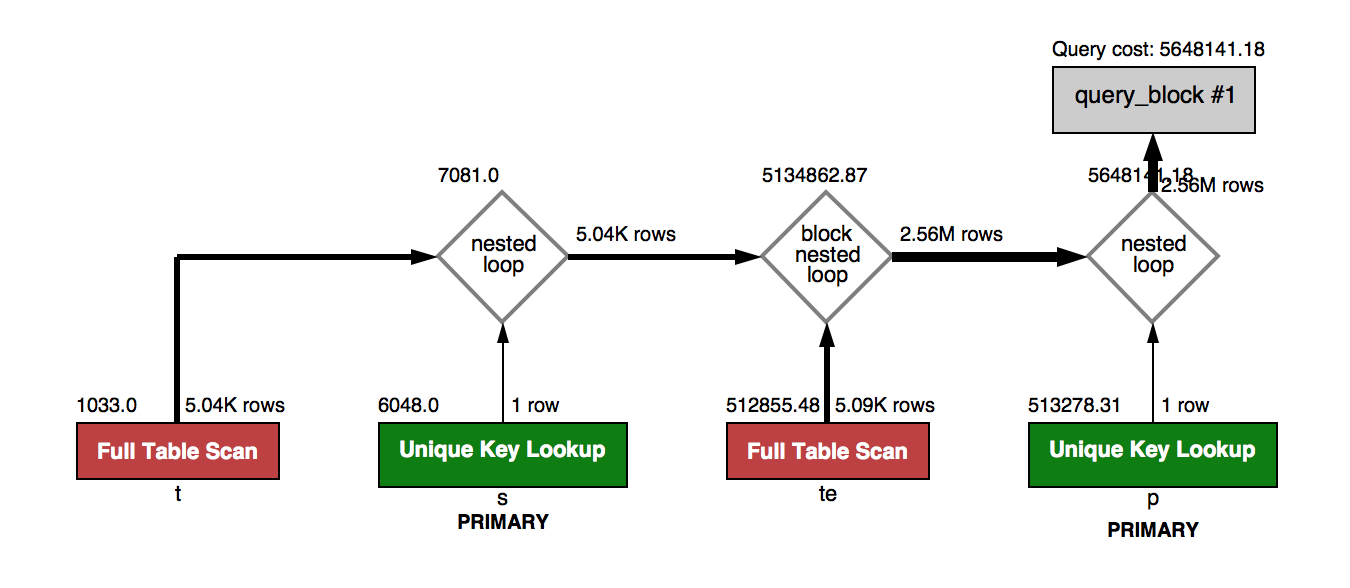
After Optimization:



* Adjusted with index for Transcript crsCode ascending called tIndsCrs
* Query adjusted to use INNER JOIN.
* Optimized duration = 0.00343800
* While the query cost was not improved significantly, the query speed was significantly quicker after optimizing with indexes and inner joins.

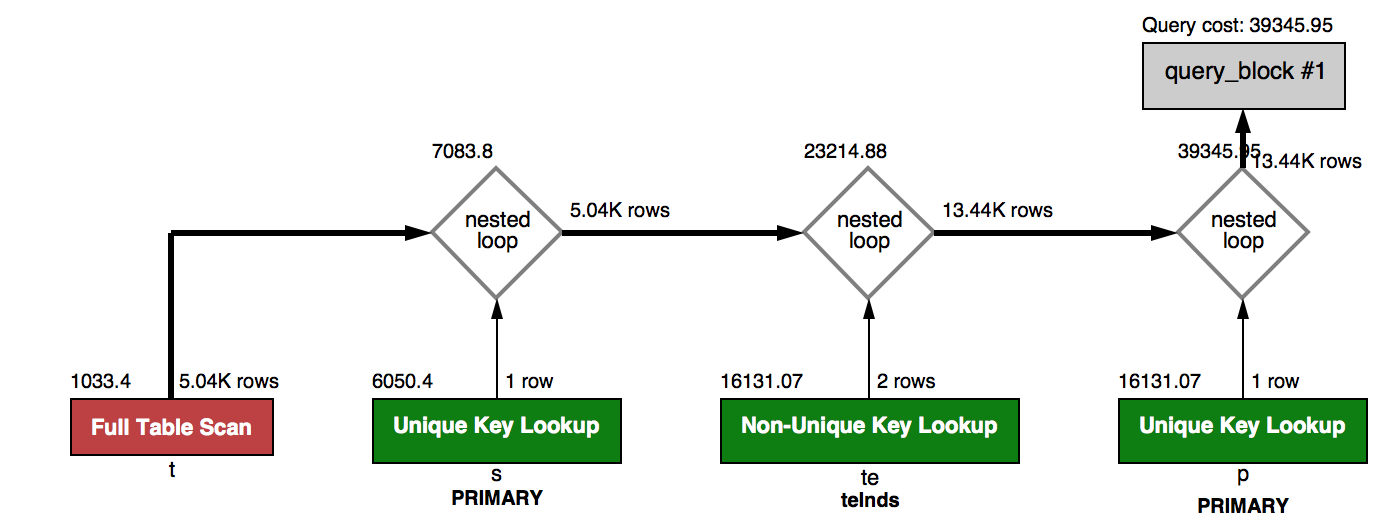
Query 4:

Before Optimization:



* Select s.name from Student s, Transcript t,Teaching te, Professor p where s.id = t.studId AND te.crsCode = t.crsCode AND te.profId = p.id AND p.name = @v5;
* Original duration = 0.01800100

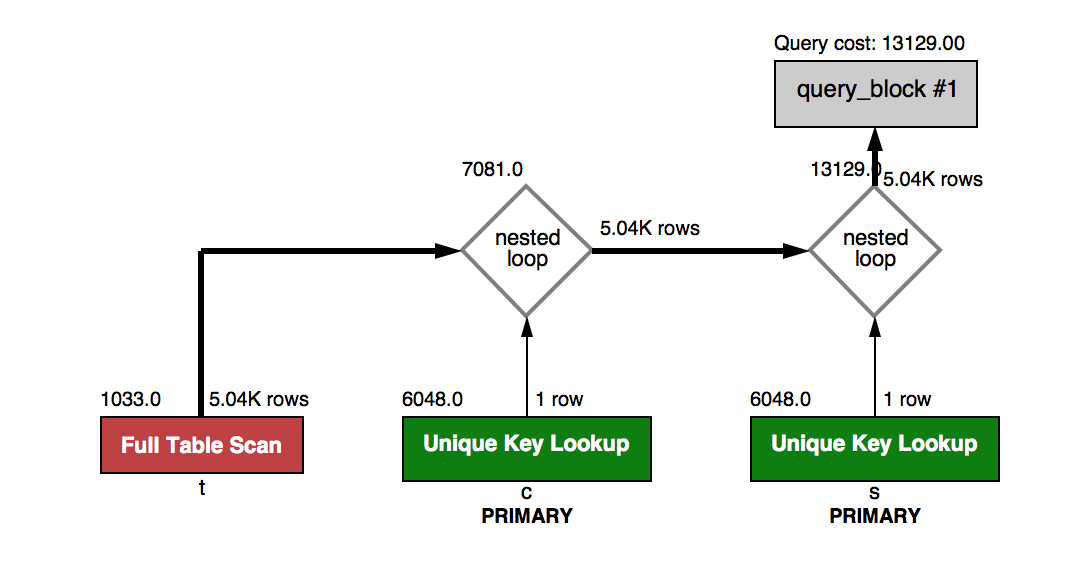
After Optimization:



* Adjusted with index for
* Transcript studId ascending – tInds
* Transcript crsCode ascending – tIndsCrs
* Teaching crsCode ascending – teInds
* Teaching profId ascending – teIndsPro
* Query adjusted to use INNER JOIN.
* Optimized duration = 0.06716400
* The query cost was cut from 5648141.18 to 39345.95. The block nested loop was adjusted to be a nested loop and therefore only accesses 2 rows instead of 5.09K rows. Even though the speed slowed, the query was performed more efficiently with indexes and inner joins.

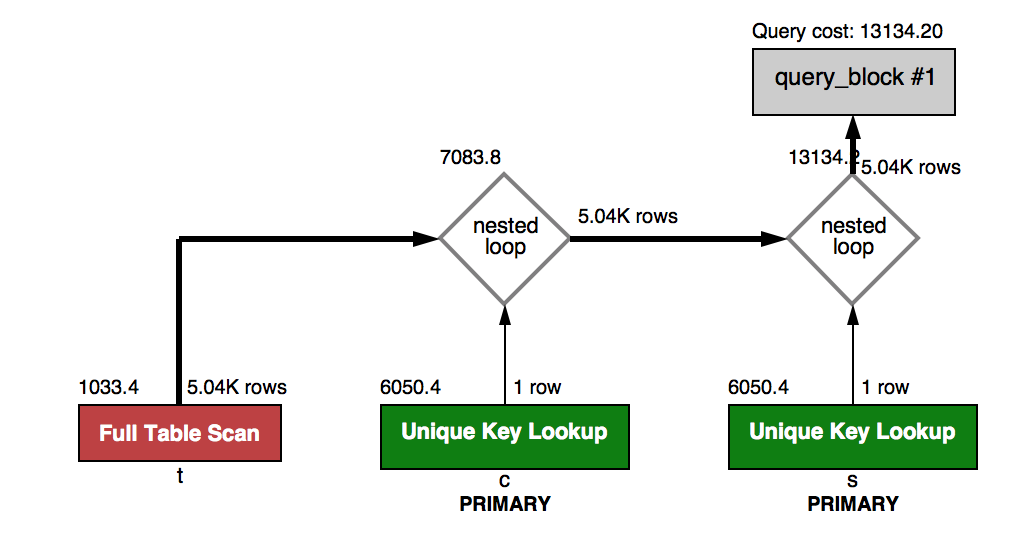
Query 5:

Before Optimization:



* Select s.name from Student s, Course c ,Transcript t where t.studId = s.id AND t.crsCode = c.crsCode AND c.deptId = @v6 AND c.deptId != @v7 ;
* Original duration = 0.00717800

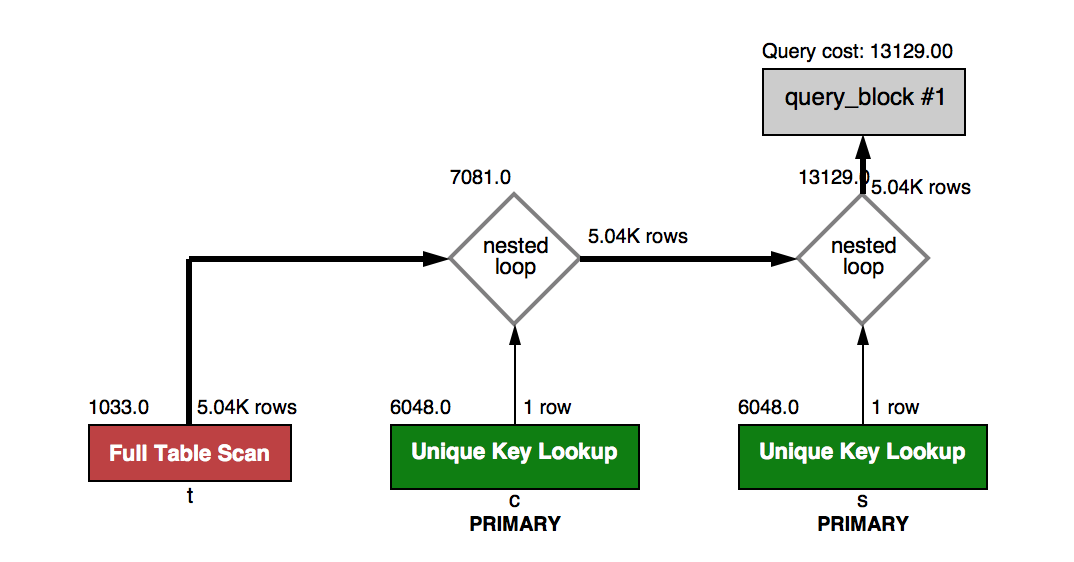
After Optimization:



* Adjusted with index for
* Transcript studId ascending – tInds
* Transcript crsCode ascending – tIndsCrs
* Course deptId ascending – cIndDep
* Query adjusted to use INNER JOIN.
* Optimized duration = 0.00740500
* There was no significant change for this query even with indexes and inner joins because nested loops and unique key lookup was already in play.

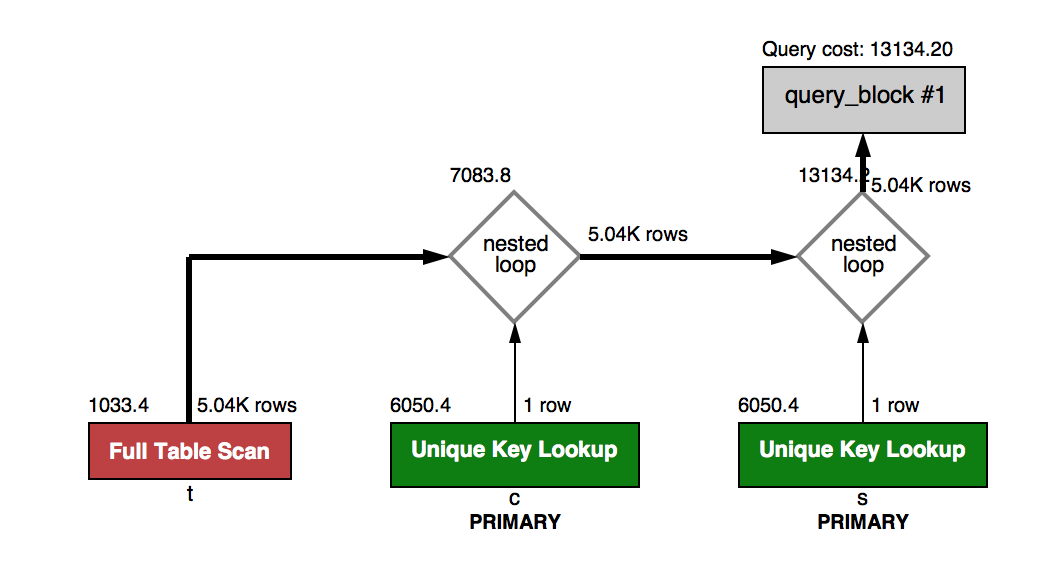
Query 6:

Before Optimization:



* Select s.name from Student s, Course c ,Transcript t where s.id = t.studId AND t.crsCode = c.crsCode AND c.deptId = @v8 ;
* Original duration = 0.00816100

After Optimization:



* Adjusted with index for
* Transcript studId ascending – tInds
* Transcript crsCode ascending – tIndsCrs
* Course deptId ascending – cIndDep
* Query adjusted to use INNER JOIN.
* Optimized duration: 0.00970600
* There was no significant change for this query even with indexes and inner joins because nested loops and unique key lookup was already in play.