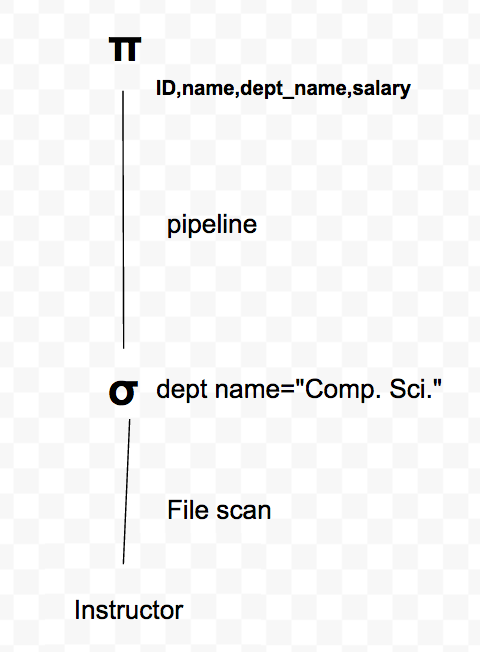
1.

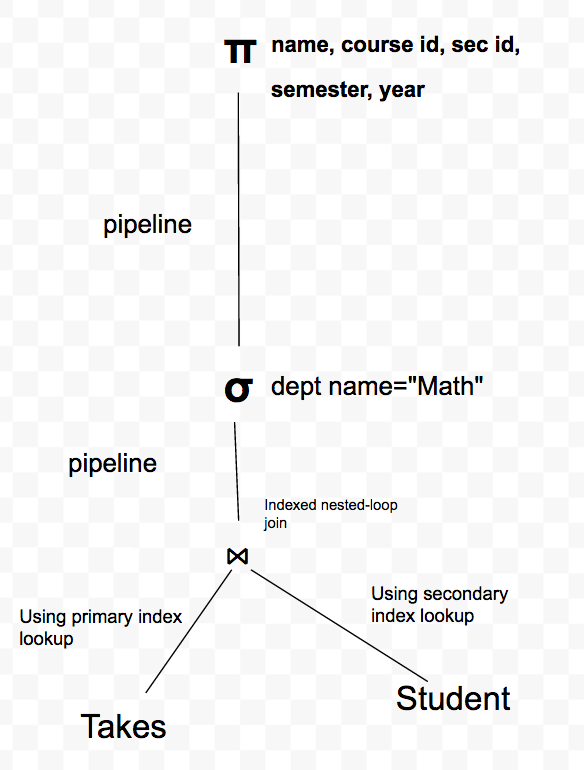
(a). Equality search by using Secondary Index on course\_id to find instructors in Computer Science department, and it uses constant value lookup.

**πID,name,dept\_name,salary(σdept\_name="Comp. Sci."**(Instructor))



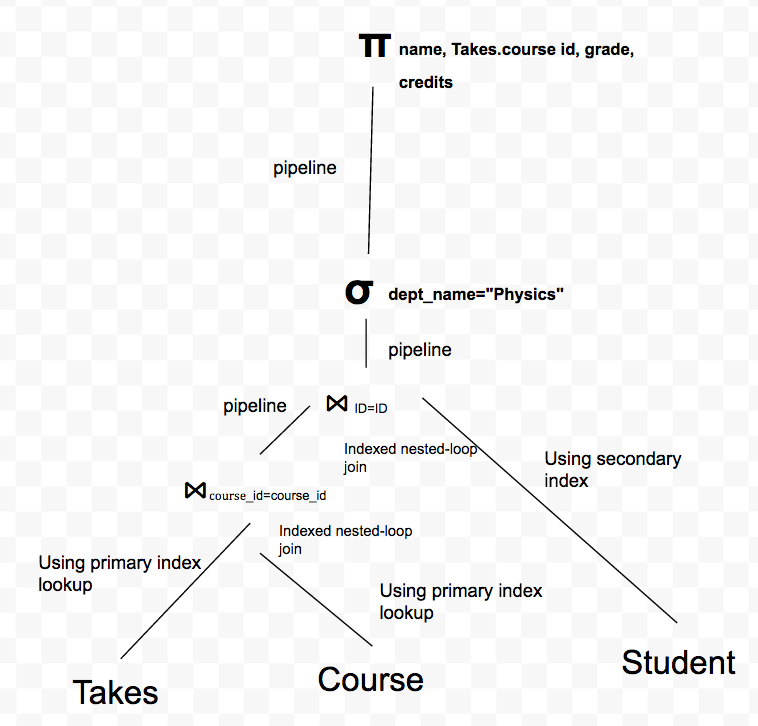
(b). Indexed nested-loop join by secondary index on Student.dept\_name, and using primary index scan on Takes, and using equality search using secondary index lookup on student&using non-key field on dept\_name to find courses which math students take, and their names.

**πname, course\_id, sec\_id, semester, year (σdept\_name="Math"**(Takes⋈Student))



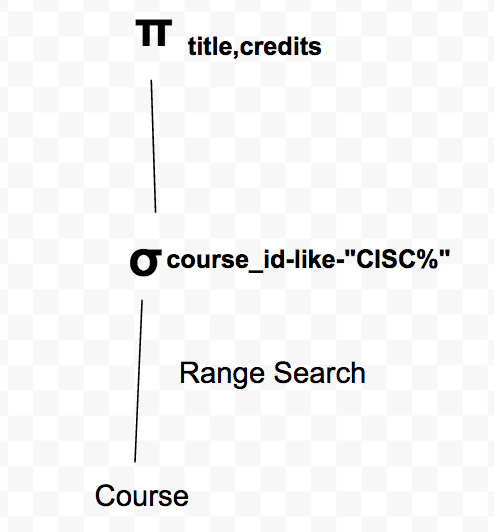
(c). Indexed nested-loop join on primary index on Takes, using equality search using secondary index lookup on Student.dept\_name, using equality search using primary index lookup on Takes and Course, but for Course there is only 1 matching record. To find these courses which physics students take and their name and their grades, and courses’ credits.

**πname, course\_id, grade, credits (σdept\_name="Physics"**(Student⋈ID=ID Takes⋈course\_id=course\_idCourse))



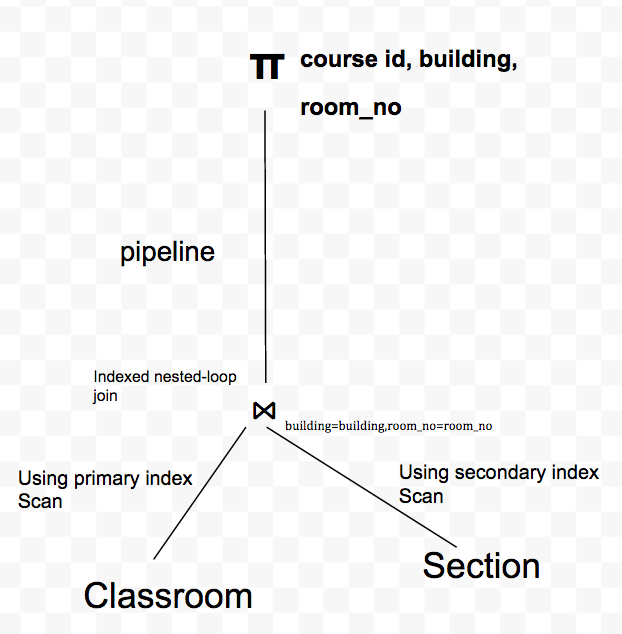
(d). Range Search Using Primary Index on Course to find CS department’s courses for their titles and credits.

**πtitle,credits(σcourse\_id-like-"CISC%"**(Course))



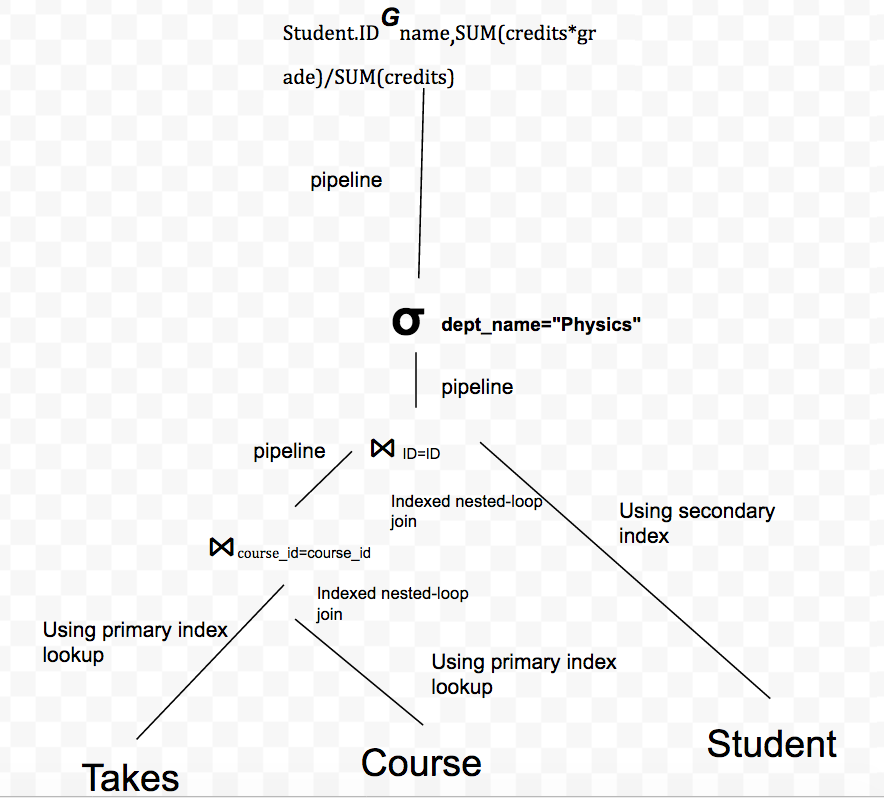
(e). Indexed nested-loop join on primary key on Classroom, Classroom is using a primary index scan on primary key, and section is using a secondary index scan on building, room\_no, to find courses’ course\_id and their building and room\_no.

**πcourse id, building, room\_no (**Classroom⋈building=building,room\_no=room\_noSection))



(f). This is aggregation by Primary Key, function on non-key attributes(grade and credits), using primary index lookup for Takes and Course, but only one record match for Course, and there is an Equality Search Using Secondary Index on Student for dept\_name to find every student’s name in Physics department and their individual GPA.

Student.ID*G*name,SUM(credits\*grade)/SUM(credits) **(σdept\_name="Physics"**(Student⋈ID=ID Takes⋈course\_id=course\_idCourse))



2.

(a).

table scan= 2000\*108=216000 bytes, 216000/4000 blocks

pipeline for dept\_name=PHYS: 1080/4000 bytes1 block

**More efficient: using secondary index on dept\_name**

(b).

table scan=55blocks

indexed nested-loop=55+2000\*(2+15)=34055 blocks

Pipeline(after join)=((28+108)\*30000)/4000=1020 blocks

Pipeline(name=”Alan Turing”)1 block

**More efficient: using sort-merge join for instructor and teaches**

(c).**From bottom to top order:**

index lookup=6+100=106 blocks

Sending (100\*116)/40003 blocks

block nested-loop=3\*40= 120 blocks

Pipeline1=3 blocks

indexed nested-loop=100\*(2+1)=300 blocks

Pipeline2=(100\*(108+8+108))/4000= 6 blocks

Pipeline31 block

**More efficient: Join Student and instructor first.**

(d). **From bottom to top order:**

Table scan for student: 541 blocks

Merge join=541+40=581 blocks

Sending (20000\*(108+8))/4000=580 blocks

Block nested loop for student, advisor, instructor: 580+580\*55=32480 blocks

Block nested loop for all: 1120\*212=237440 blocks

Pipeline1=(252\*30000)/4000 =1890 blocks

Pipeline2=252/4000= 1 block

**More efficient: Apply σstudent.name=”** **Rosalind Franklin” Before the join of student and advisor.**