The following is the query I discussed in class, using the subquery w/ set operations. The second uses a join across three tables to substitute for the correlated subquery (on correlation variable student.id). In practice, joins are a bit easier for the query executor to optimize, but the first form is probably more clear.

Approaches using joins and subqueries are often equivalent. A 'with' clause doesn't have the same power unless you use a join in the with clause.

(1)

```sql
SELECT DISTINCT id,name
FROM student
WHERE NOT EXISTS  ((SELECT course_id
FROM course
WHERE course.dept_name='Finance')
EXCEPT
(SELECT course_id
FROM takes
WHERE takes.id=student.id));
```

(2)

```sql
WITH coursefin(ccnt) AS
(SELECT COUNT(DISTINCT course_id)
FROM course
WHERE dept_name = 'Finance'),
takesfin(sid,name,ccnt) AS
(SELECT t.id, name, COUNT(DISTINCT c.course_id)
FROM takes t,student s,course c
WHERE t.course_id=c.course_id
AND t.id=s.id
AND c.dept_name='Finance'
GROUP BY t.id,name)

SELECT name
FROM coursefin c,takesfin t
WHERE c.ccnt = t.ccnt;
```