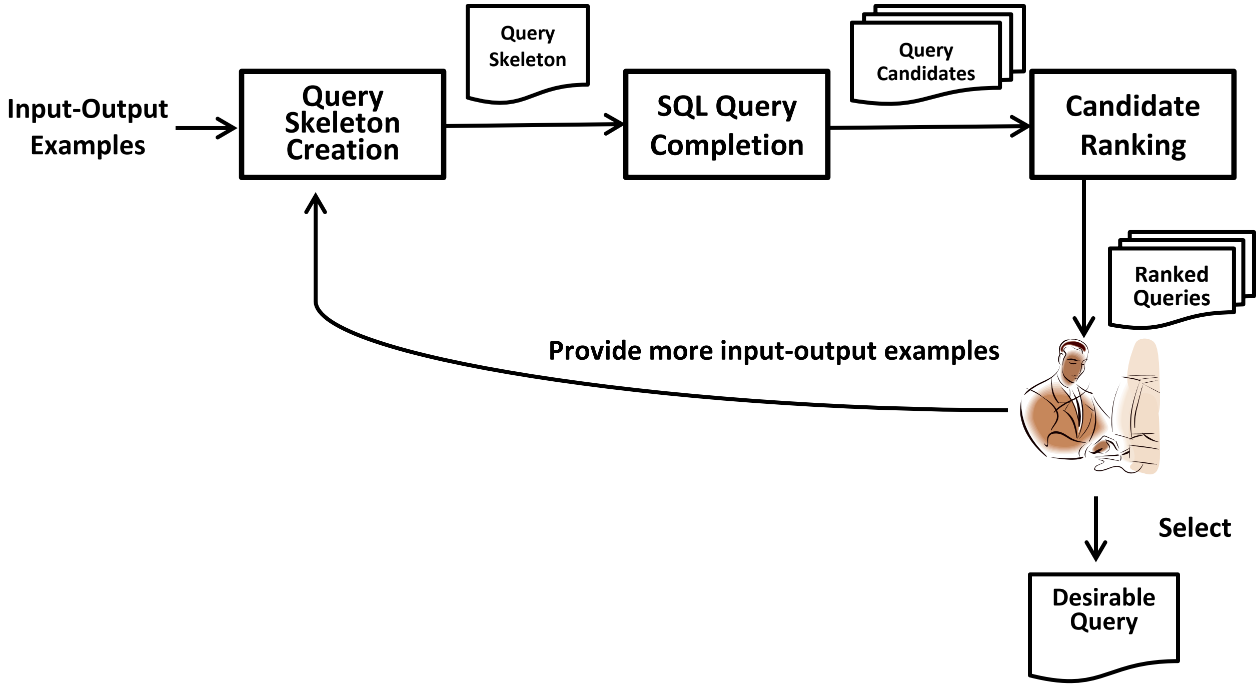
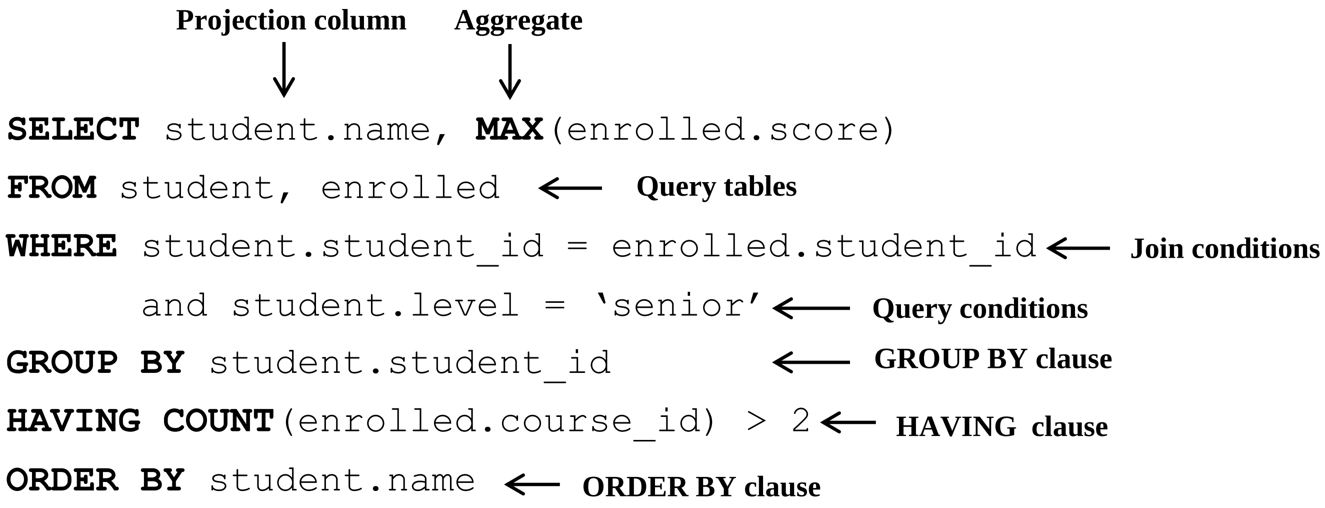

Input-Output Examples
Query Skeleton Creation
Query Skeleton
SQL Query Completion
Query Candidates
Candidate Ranking
Ranked Queries
Provide more input-output examples
Select
Desirable Query



```
SELECT student.name, MAX(enrolled.score)
FROM student, enrolled
WHERE student.student_id = enrolled.student_id
      and student.level = 'senior'
GROUP BY student.student_id
HAVING COUNT(enrolled.course_id) > 2
ORDER BY student.name
```

```
select    Student.Student_name, <Aggregation>
from      student, enrolled
where     student.Student_key = enrolled.Student_key
          and  <Conditions>
group by  Student.Student_name
having    <Conditions>
```

Fgg

ff

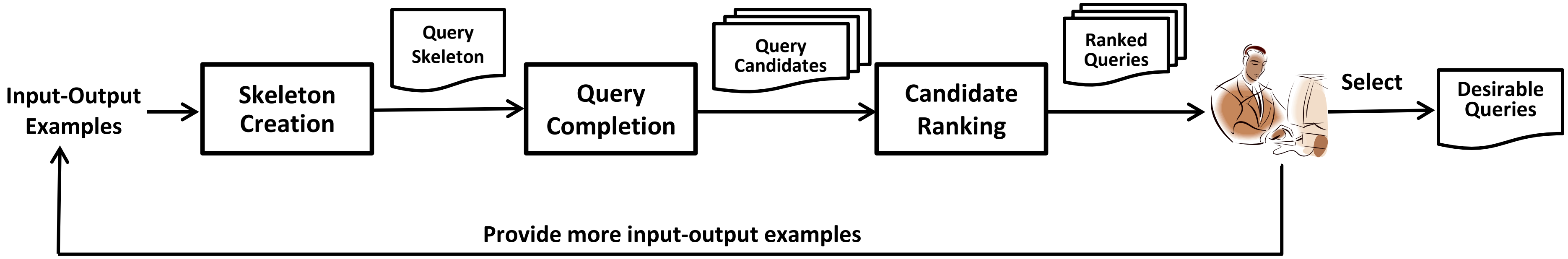
name	score
Bob	4
Dan	5
Jim	2

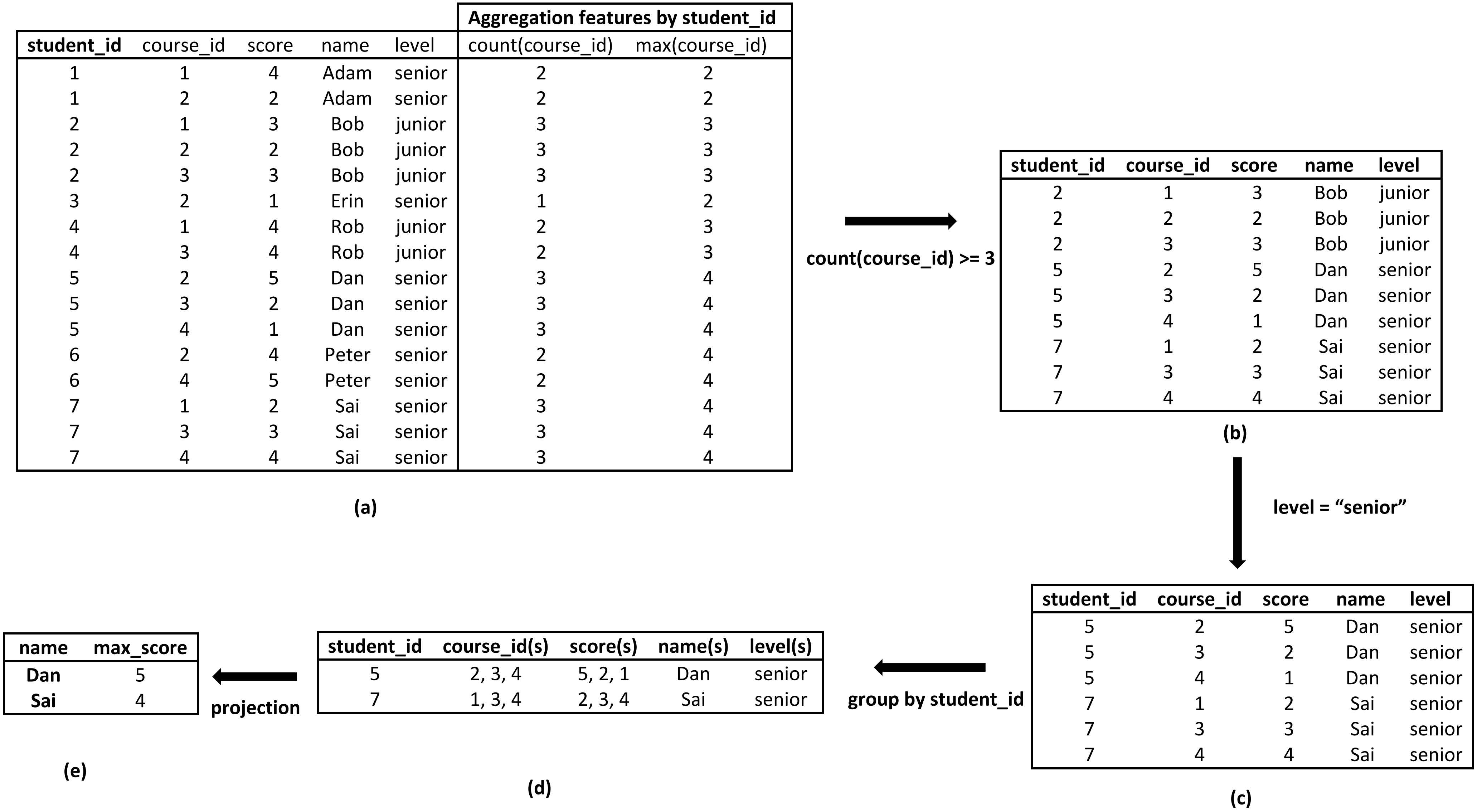
name
Bob
Dan

(a) The input table: student

(b) The output table

```
1. select name from student where score > 2
2. select name from student where name = 'Bob'
                                  or name = 'Dan'
```


Input-Output Examples
Skeleton Creation
Query Skeleton
Query Completion
Query Candidates
Candidate Ranking
Ranked Queries
Select
Desirable Queries
Provide more input-output examples



					Aggregation features by student_id	
student_id	course_id	score	name	level	count(course_id)	max(course_id)
1	1	4	Adam	senior	2	2
1	2	2	Adam	senior	2	2
2	1	3	Bob	junior	3	3
2	2	2	Bob	junior	3	3
2	3	3	Bob	junior	3	3
3	2	1	Erin	senior	1	2
4	1	4	Rob	junior	2	3
4	3	4	Rob	junior	2	3
5	2	5	Dan	senior	3	4
5	3	2	Dan	senior	3	4
5	4	1	Dan	senior	3	4
6	2	4	Peter	senior	2	4
6	4	5	Peter	senior	2	4
7	1	2	Sai	senior	3	4
7	3	3	Sai	senior	3	4
7	4	4	Sai	senior	3	4

(a)

count(course_id) >= 3

student_id	**course_id**	**score**	**name**	**level**
2	1	3	Bob	junior
2	2	2	Bob	junior
2	3	3	Bob	junior
5	2	5	Dan	senior
5	3	2	Dan	senior
5	4	1	Dan	senior
7	1	2	Sai	senior
7	3	3	Sai	senior
7	4	4	Sai	senior

(b)

level = “senior”

student_id	**course_id**	**score**	**name**	**level**
5	2	5	Dan	senior
5	3	2	Dan	senior
5	4	1	Dan	senior
7	1	2	Sai	senior
7	3	3	Sai	senior
7	4	4	Sai	senior

(c)

group by student_id

student_id	**course_id(s)**	**score(s)**	**name(s)**	**level(s)**
5	2, 3, 4	5, 2, 1	Dan	senior
7	1, 3, 4	2, 3, 4	Sai	senior

(d)

projection

name	**max_score**
Dan	5
Sai	4

(e)

An input table

C1	C2
2	4
2	1
2	1
1	1

Aggregation Features

Group by C1					Group by C2				
Count	Count Distinct	Min	Max	Avg	Count	Count Distinct	Min	Max	Avg
3	2	1	4	2	1	1	2	2	2
3	2	1	4	2	3	2	1	2	5/3
3	2	1	4	2	3	2	1	2	5/3
4	2	1	1	1	3	2	1	2	5/3

Comparison Features

C1 = C2	C1 < C2	C1 > C2
0	1	0
0	0	1
0	0	1
1	0	0

Column1	Column2	Column3	Column 4
101	2001	3020	01-01-11
101	2001	3002	02-01-11
101	2001	3001	03-01-11
102	2002	3002	01-01-11

Column1	Column2	Column 3
20011	2001	200131
20012	2001	200132
20013	2001	200133

Column1	Column 2
20011	Site
20012	Site
20013	Site

```
select min(T1.Column1), T2.Column3,
       min(T1.Column4), min(T3.Column2)
from T1, T2, T3
where T1.Column2 = T2.Column2
      and T2.Column1 = T3.Column1
group by T2.Column3
```

101	200131	01-01-11	Site
101	200132	01-01-11	Site
101	200133	01-01-11	Site

T3 (right)

(b) A SQL query inferred by SQLSythensizer

(c) The output table

student_id	name	level
1	Adam	senior
2	Bob	junior
3	Erin	senior
4	Rob	junior
5	Dan	senior
6	Peter	senior
7	Sai	senior

student_id	course_id	score
1	1	4
1	2	2
2	1	3
2	2	2
2	3	3
3	2	1
4	1	4
4	3	4
5	2	5
5	3	2
5	4	1
6	2	4
6	4	5
7	1	2
7	3	3
7	4	4

(a) Two input tables: student (Left) and enrolled (Right)

```
SELECT student.name, MAX(enrolled.score)
FROM student, enrolled
WHERE student.student_id = enrolled.student_id
      and student.level = ‘senior’
GROUP BY student.student_id
HAVING COUNT(enrolled.course_id) > 2
```

(b) A SQL query inferred by SQLSynthesizer

name	max_score
Dan	5
Sai	5

(c) An output table