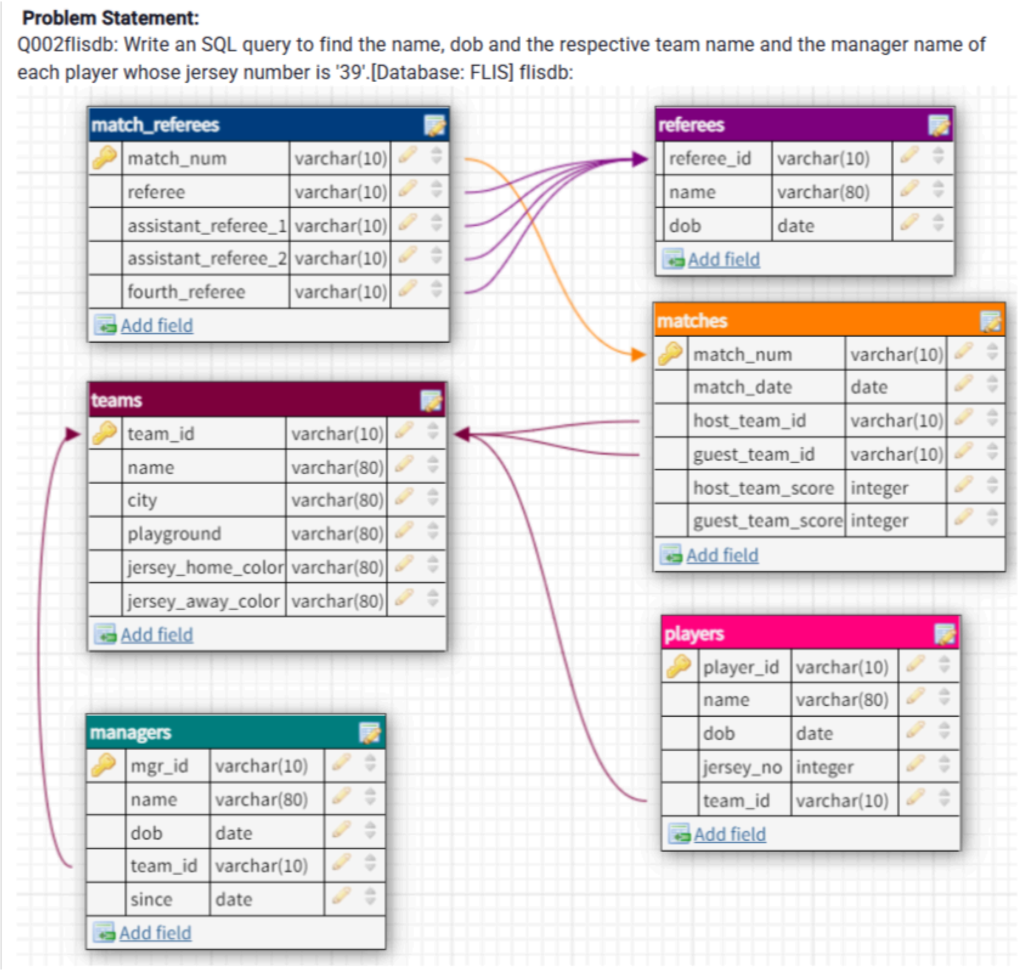
OPPE

OPPE SQL Questions



Problem Statement:
Q002flisdb: Write an SQL query to find the name, dob and the respective team name and the manager name of each player whose jersey number is '39'.[Database: FLIS] flisdb:

match_referees	
match_num	varchar(10)
referee	varchar(10)
assistant_referee_1	varchar(10)
assistant_referee_2	varchar(10)
fourth_referee	varchar(10)
Add field	

referees	
referee_id	varchar(10)
name	varchar(80)
dob	date
Add field	

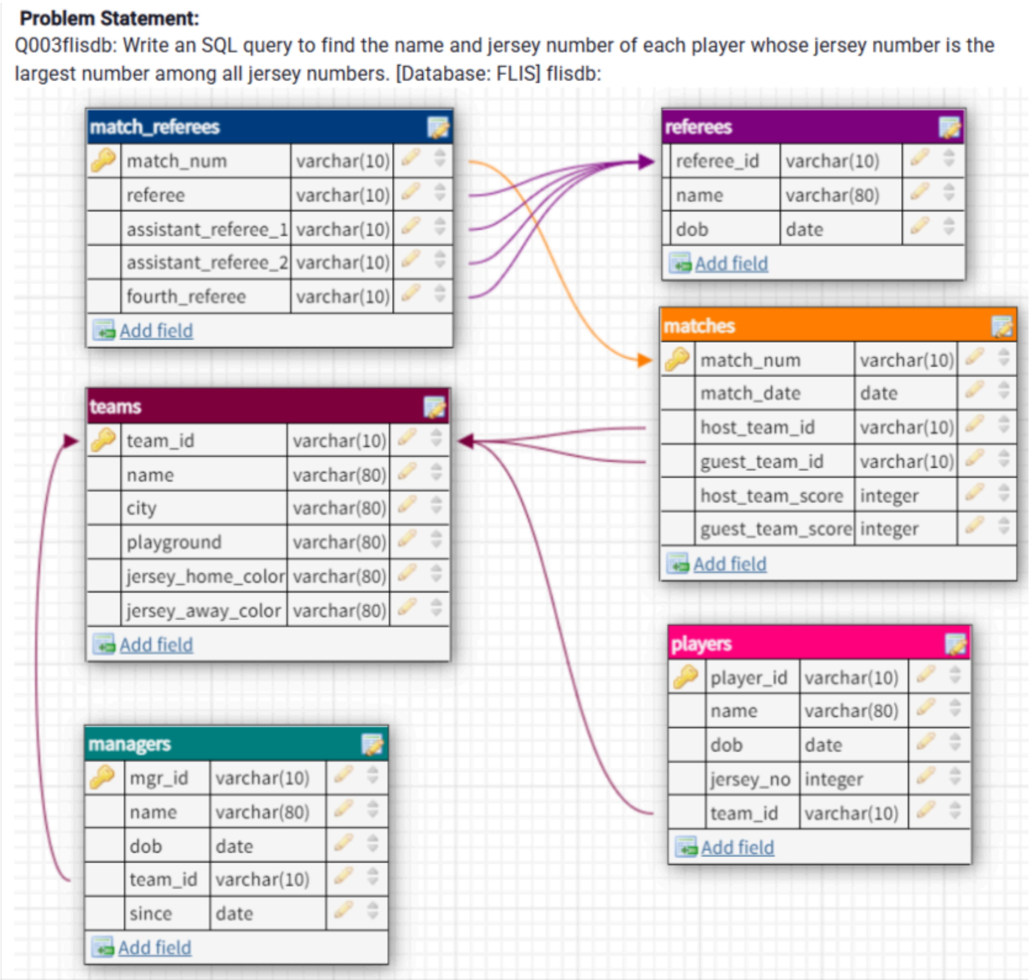
matches	
match_num	varchar(10)
match_date	date
host_team_id	varchar(10)
guest_team_id	varchar(10)
host_team_score	integer
guest_team_score	integer
Add field	

teams	
team_id	varchar(10)
name	varchar(80)
city	varchar(80)
playground	varchar(80)
jersey_home_color	varchar(80)
jersey_away_color	varchar(80)
Add field	

players	
player_id	varchar(10)
name	varchar(80)
dob	date
jersey_no	integer
team_id	varchar(10)
Add field	

managers	
mgr_id	varchar(10)
name	varchar(80)
dob	date
team_id	varchar(10)
since	date
Add field	

```
select p.name as player_name, p.dob, t.name as team_name, m.name as manager_name from players p
join teams t on p.team_id = t.team_id
join managers m on t.team_id = m.team_id
where jersey_no = 30;
```



Problem Statement:

Q003flisdb: Write an SQL query to find the name and jersey number of each player whose jersey number is the largest number among all jersey numbers. [Database: FLIS] flisdb:

match_referees	
match_num	varchar(10)
referee	varchar(10)
assistant_referee_1	varchar(10)
assistant_referee_2	varchar(10)
fourth_referee	varchar(10)

referees	
referee_id	varchar(10)
name	varchar(80)
dob	date

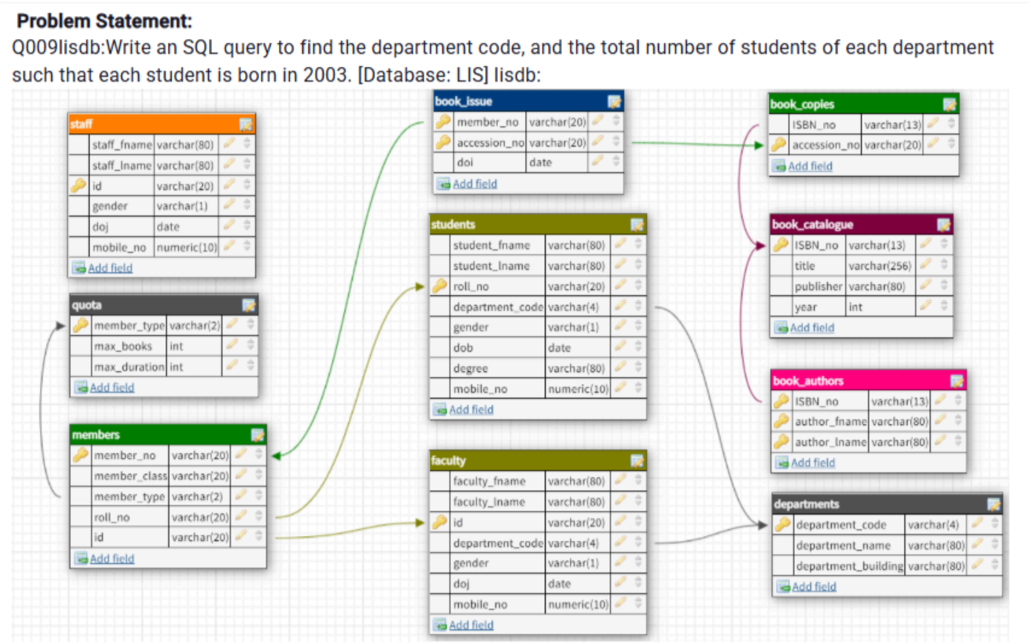
matches	
match_num	varchar(10)
match_date	date
host_team_id	varchar(10)
guest_team_id	varchar(10)
host_team_score	integer
guest_team_score	integer

teams	
team_id	varchar(10)
name	varchar(80)
city	varchar(80)
playground	varchar(80)
jersey_home_color	varchar(80)
jersey_away_color	varchar(80)

players	
player_id	varchar(10)
name	varchar(80)
dob	date
jersey_no	integer
team_id	varchar(10)

managers	
mgr_id	varchar(10)
name	varchar(80)
dob	date
team_id	varchar(10)
since	date

```
select name, jersey_no from players
where jersey_no = (select max(jersey_no) from players)
```



Problem Statement:

Q009lisdb:Write an SQL query to find the department code, and the total number of students of each department such that each student is born in 2003. [Database: LIS] lisdb:

	staff	
	staff_fname	varchar(80)
	staff_lname	varchar(80)
🔑	id	varchar(20)
	gender	varchar(1)
	doj	date
	mobile_no	numeric(10)

	quota	
🔑	member_type	varchar(2)
	max_books	int
	max_duration	int

	members	
🔑	member_no	varchar(20)
	member_class	varchar(20)
	member_type	varchar(2)
	roll_no	varchar(20)
	id	varchar(20)

	book_issue	
🔑	member_no	varchar(20)
🔑	accession_no	varchar(20)
	doi	date

	students	
	student_fname	varchar(80)
	student_lname	varchar(80)
🔑	roll_no	varchar(20)
	department_code	varchar(4)
	gender	varchar(1)
	dob	date
	degree	varchar(80)
	mobile_no	numeric(10)

	faculty	
	faculty_fname	varchar(80)
	faculty_lname	varchar(80)
🔑	id	varchar(20)
	department_code	varchar(4)
	gender	varchar(1)
	doj	date
	mobile_no	numeric(10)

	book_copies	
	ISBN_no	varchar(13)
🔑	accession_no	varchar(20)

	book_catalogue	
🔑	ISBN_no	varchar(13)
	title	varchar(256)
	publisher	varchar(80)
	year	int

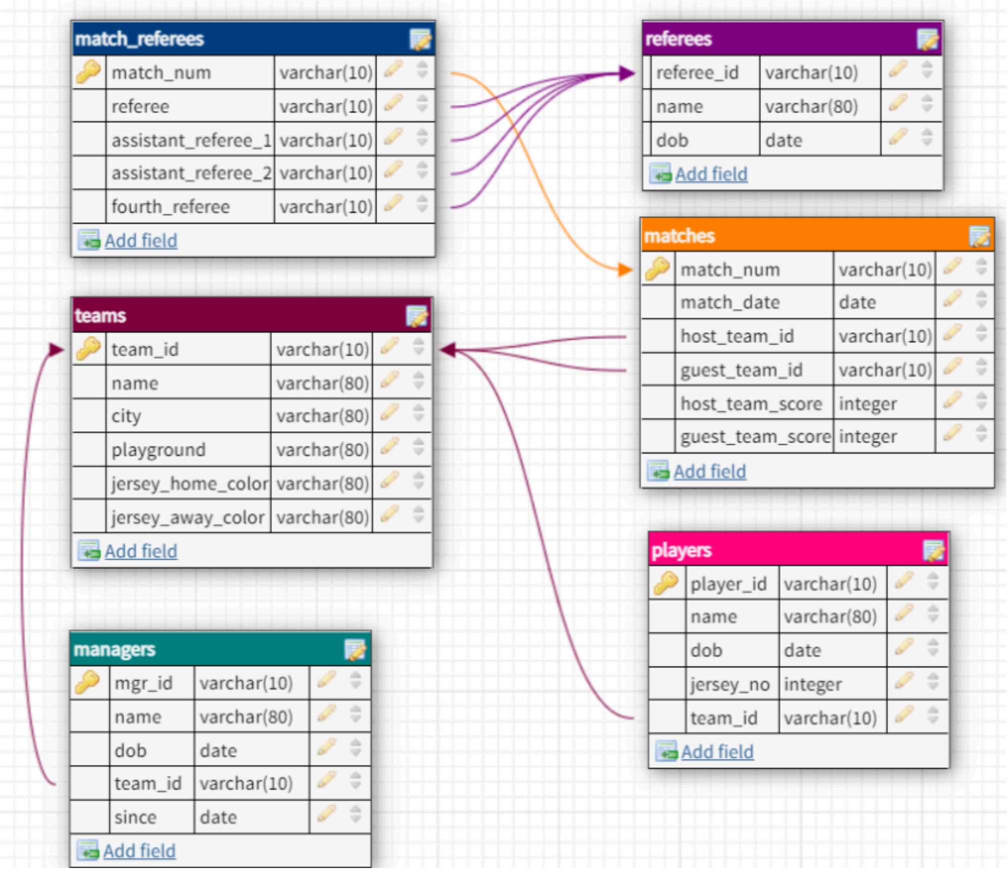
	book_authors	
🔑	ISBN_no	varchar(13)
🔑	author_fname	varchar(80)
🔑	author_lname	varchar(80)

	departments	
🔑	department_code	varchar(4)
	department_name	varchar(80)
	department_building	varchar(80)

```
SELECT department_code, count(*) as total_students from students
where extract(year from dob) = 2003
group by department_code
```

Problem Statement:

Q003flisdb:Write an SQL query to find the names of all players of team 'Amigos'.[Database: FLIS] flisdb:



match_referees

match_num	varchar(10)
referee	varchar(10)
assistant_referee_1	varchar(10)
assistant_referee_2	varchar(10)
fourth_referee	varchar(10)

referees

referee_id	varchar(10)
name	varchar(80)
dob	date

matches

match_num	varchar(10)
match_date	date
host_team_id	varchar(10)
guest_team_id	varchar(10)
host_team_score	integer
guest_team_score	integer

teams

team_id	varchar(10)
name	varchar(80)
city	varchar(80)
playground	varchar(80)
jersey_home_color	varchar(80)
jersey_away_color	varchar(80)

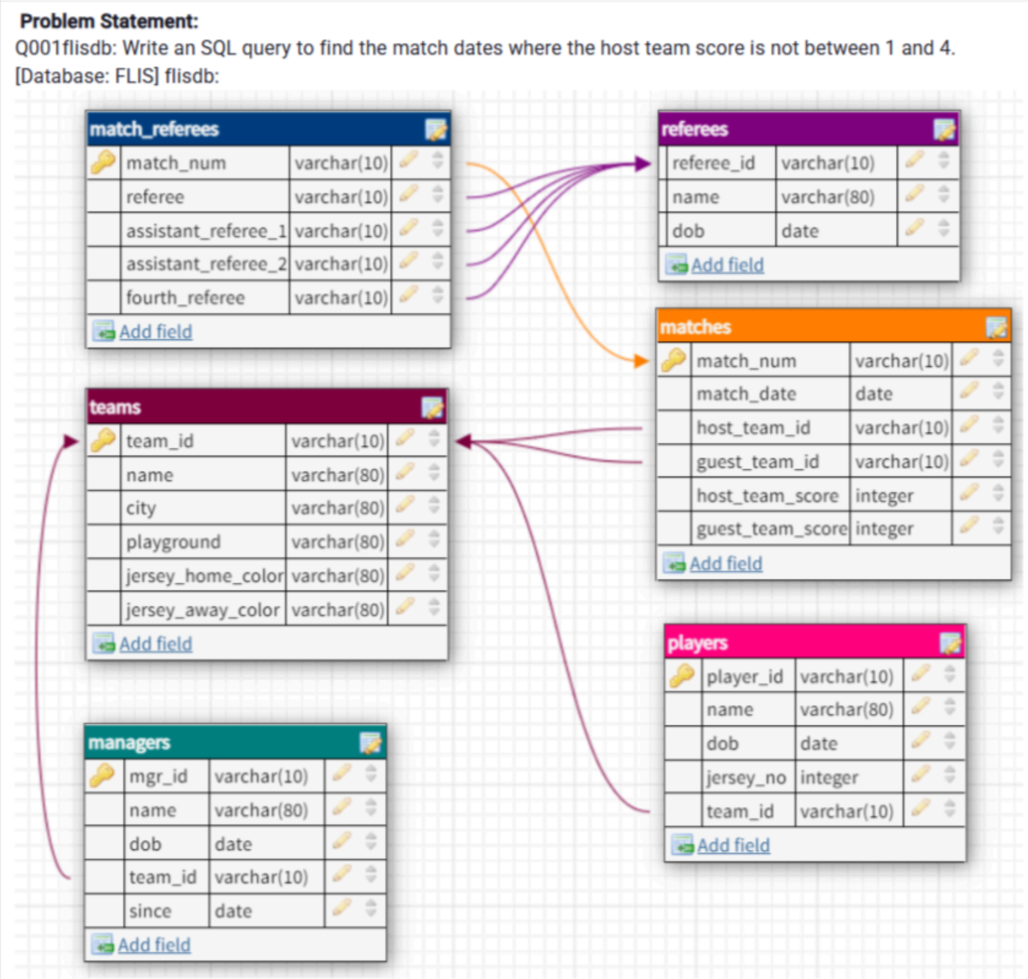
players

player_id	varchar(10)
name	varchar(80)
dob	date
jersey_no	integer
team_id	varchar(10)

managers

mgr_id	varchar(10)
name	varchar(80)
dob	date
team_id	varchar(10)
since	date

```
select p.name as player_name from players p
join teams t on p.team_id = t.team_id
where t.name = 'Amigos'
```



Problem Statement:

Q001flisdb: Write an SQL query to find the match dates where the host team score is not between 1 and 4.
[Database: FLIS] flisdb:

match_referees

🔑	match_num	varchar(10)
	referee	varchar(10)
	assistant_referee_1	varchar(10)
	assistant_referee_2	varchar(10)
	fourth_referee	varchar(10)

referees

	referee_id	varchar(10)
	name	varchar(80)
	dob	date

matches

🔑	match_num	varchar(10)
	match_date	date
	host_team_id	varchar(10)
	guest_team_id	varchar(10)
	host_team_score	integer
	guest_team_score	integer

teams

🔑	team_id	varchar(10)
	name	varchar(80)
	city	varchar(80)
	playground	varchar(80)
	jersey_home_color	varchar(80)
	jersey_away_color	varchar(80)

players

🔑	player_id	varchar(10)
	name	varchar(80)
	dob	date
	jersey_no	integer
	team_id	varchar(10)

managers

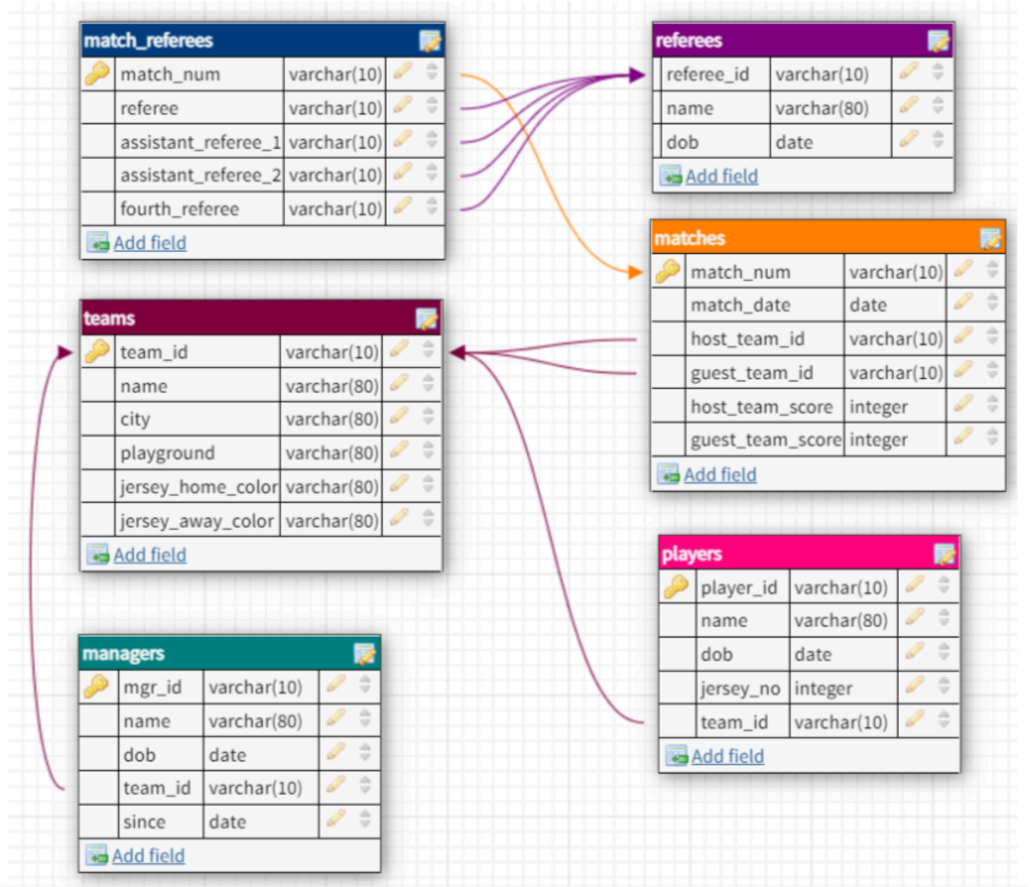
🔑	mgr_id	varchar(10)
	name	varchar(80)
	dob	date
	team_id	varchar(10)
	since	date

```
select match_date from matches
where host_team_score not between 1 and 4;
```

Problem Statement:

Q003flisdb: Write an SQL query to find the player IDs of the players whose name starts with 'K'.[Database: FLIS]

flisdb:



match_referees	
match_num	varchar(10)
referee	varchar(10)
assistant_referee_1	varchar(10)
assistant_referee_2	varchar(10)
fourth_referee	varchar(10)

referees	
referee_id	varchar(10)
name	varchar(80)
dob	date

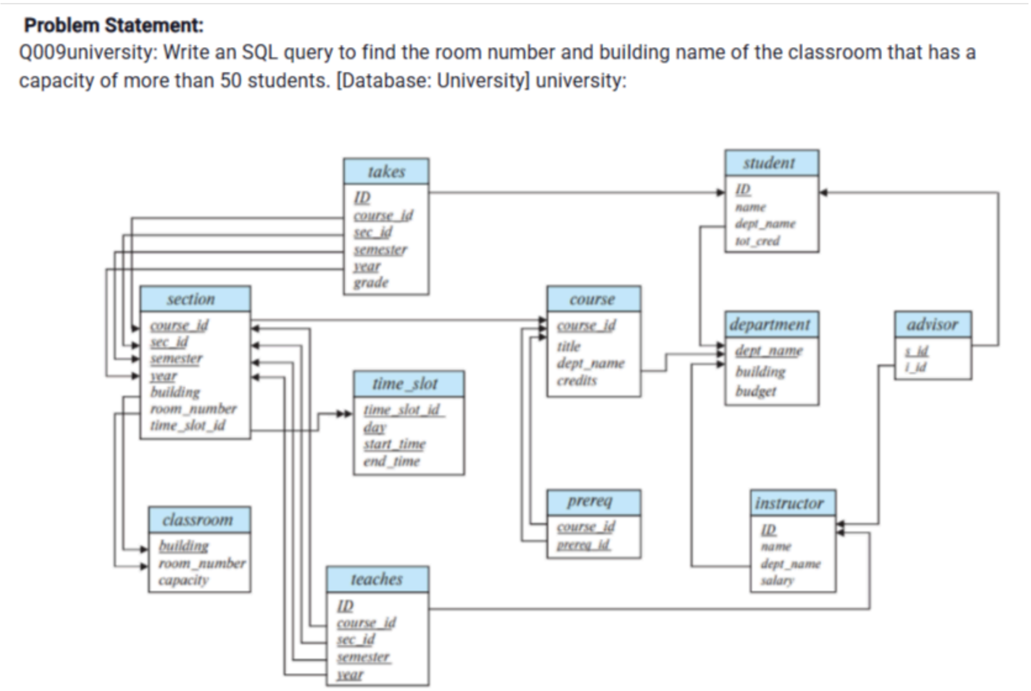
matches	
match_num	varchar(10)
match_date	date
host_team_id	varchar(10)
guest_team_id	varchar(10)
host_team_score	integer
guest_team_score	integer

teams	
team_id	varchar(10)
name	varchar(80)
city	varchar(80)
playground	varchar(80)
jersey_home_color	varchar(80)
jersey_away_color	varchar(80)

players	
player_id	varchar(10)
name	varchar(80)
dob	date
jersey_no	integer
team_id	varchar(10)

managers	
mgr_id	varchar(10)
name	varchar(80)
dob	date
team_id	varchar(10)
since	date

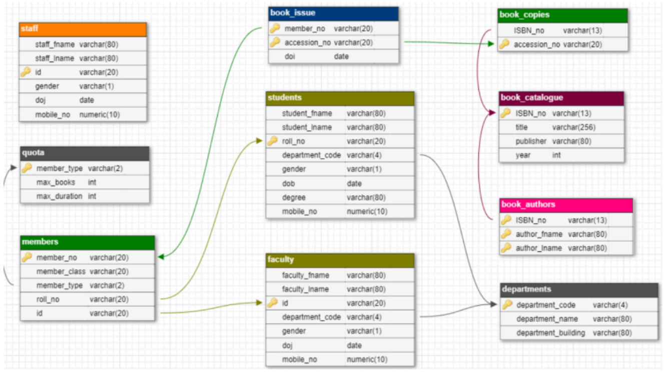
```
select player_id from players
where name like 'K%'
```

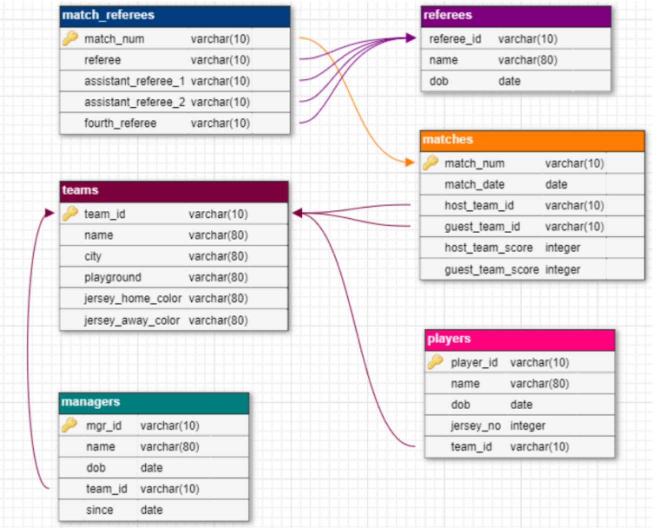


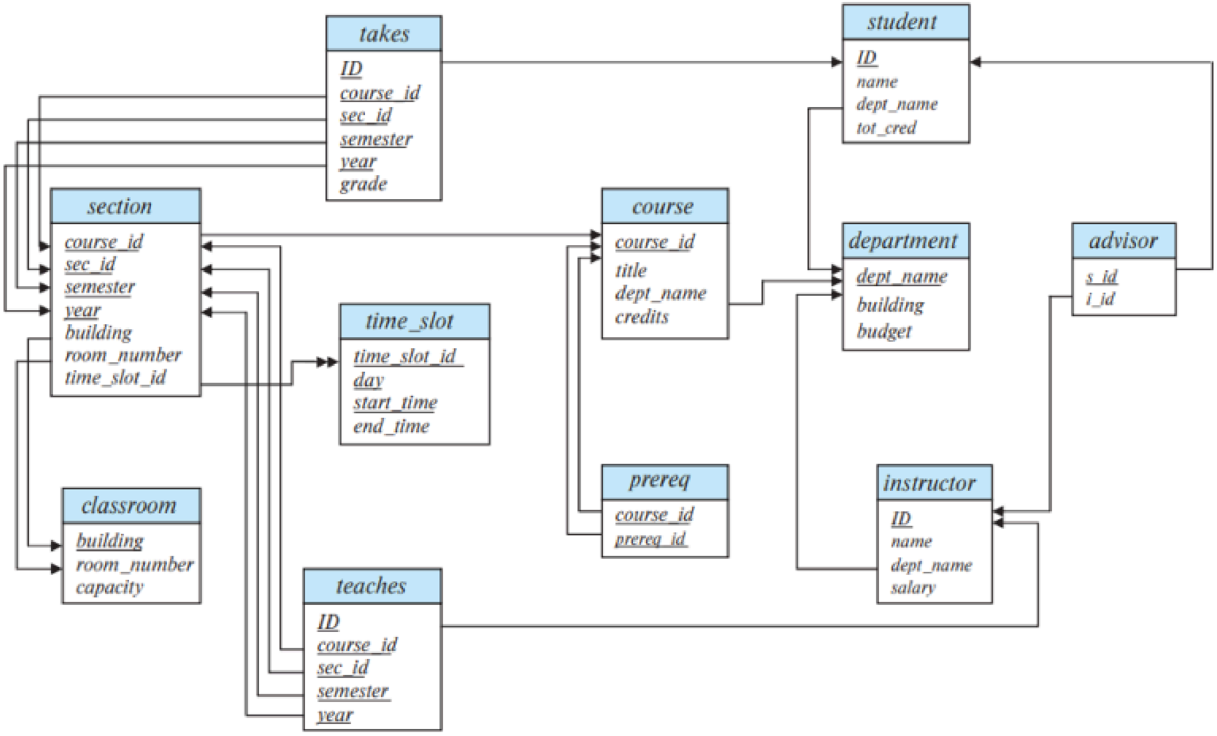
Problem Statement:

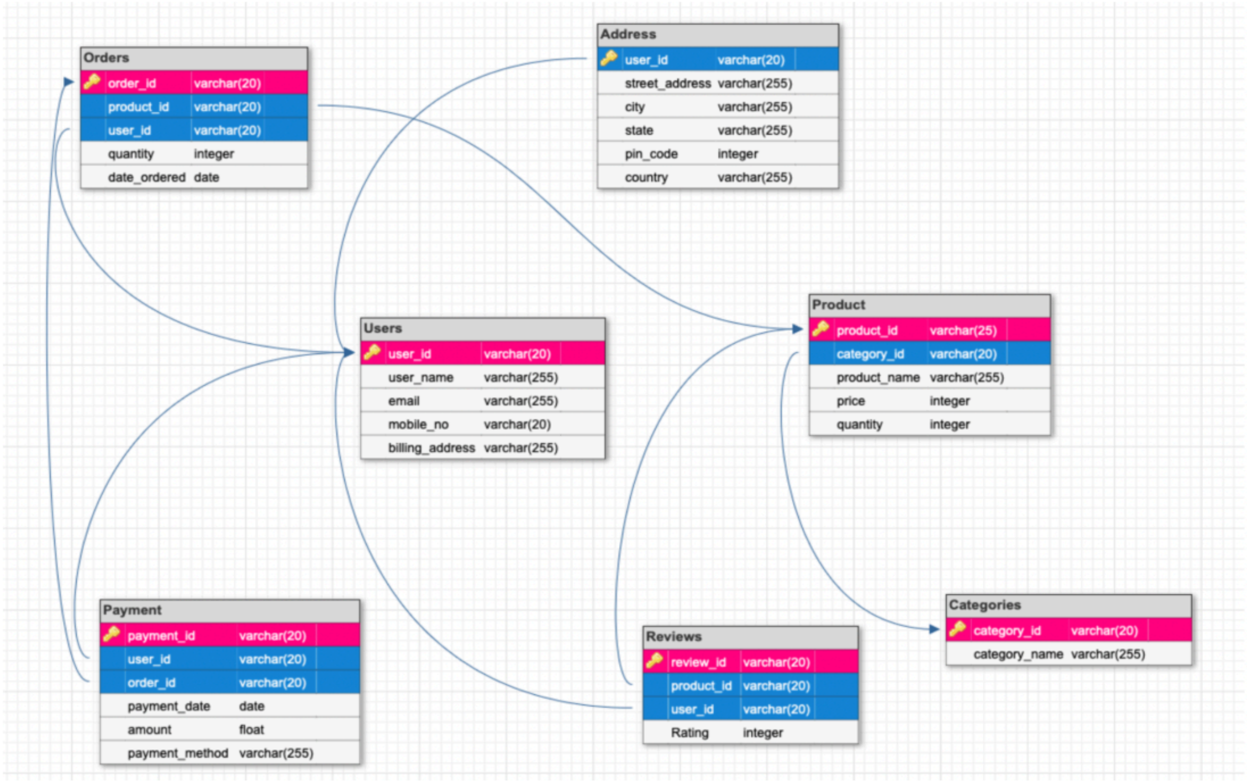
Q009university: Write an SQL query to find the room number and building name of the classroom that has a capacity of more than 50 students. [Database: University] university:

```sql
select building, room_number from classroom
where capacity > 50
```




match_referees
match_num varchar(10)
referee varchar(10)
assistant_referee_1 varchar(10)
assistant_referee_2 varchar(10)
fourth_referee varchar(10)
referees
referee_id varchar(10)
name varchar(80)
dob date
matches
match_num varchar(10)
match_date date
host_team_id varchar(10)
guest_team_id varchar(10)
host_team_score integer
guest_team_score integer
teams
team_id varchar(10)
name varchar(80)
city varchar(80)
playground varchar(80)
jersey_home_color varchar(80)
jersey_away_color varchar(80)
players
player_id varchar(10)
name varchar(80)
dob date
jersey_no integer
team_id varchar(10)
managers
mgr_id varchar(10)
name varchar(80)
dob date
team_id varchar(10)
since date


takes
ID
course_id
sec_id
semester
year
grade
student
ID
name
dept_name
tot_cred
section
course_id
sec_id
semester
year
building
room_number
time_slot_id
course
course_id
title
dept_name
credits
department
dept_name
building
budget
advisor
s_id
i_id
time_slot
time_slot_id
day
start_time
end_time
prereq
course_id
prereq_id
instructor
ID
name
dept_name
salary
classroom
building
room_number
capacity
teaches
ID
course_id
sec_id
semester
year



Find the first names and last names of authors, having the author’s first name as a single

Character. (Database: LIS)

```
select author_fname, author_lname from book_authors
where length(author_fname) = 1
```

Find the titles and publishers of all books, except the ones published in year ‘2015’ or

‘2017’. (Database: LIS)

```
select title,publisher,year from book_catalogue
where year not in (2015,2017)
```

```
select title,publisher,year from book_catalogue
where year <> 2015 and year <> 2017
```

Find the first names and last names of the students whose birthday is in May 2002 or

in May 2003. (Database: LIS)

```
select student_fname, student_lname,dob from students
where (extract(year from dob) = 2002 and extract(month from dob)= 5) or
(extract (year from dob) = 2003 and extract(month from dob)=5);
```

Find out the total number of members in the UG with alias name or column header as ‘total member’. (Database: LIS)

```
select member_type, count(member_type) as total_member from members
where member_type = 'UG'
group by member_type
```

Find out the number of female students in each department. (Database: LIS)

```
-- for sort by depart_name
select d.department_name, count(s.gender) as no_of_female_student from stu
dents s
join departments d on s.department_code = d.department_code
where s.gender = 'F'
group by d.department_name
-- for sort by depart_code
select department_code, count(gender) as no_of_female_student from student
s
where gender = 'F'
group by department_code
```

Find the names and date-of-births of those managers who have joined in years 2019 and

2020. (Database: FLIS)

```
/*Q1*/ select name as manager_name, dob from managers
```

```
where extract (year from since) in (2019,2020)
```

Find the names of all those teams where the last name of the team starts with the letter S. Use the name attribute of the teams table to answer this question. (Database: FLIS)

```
select name as team_name from teams
where name like '_% S%'
```

Print the name, dept name, salary of instructors department wise, where the department name should be sorted in descending order and within each department, the instructor names should be sorted in ascending order of salary within that department. (Database: University)

```
select name as instructor, dept_name as department_name, salary from instru
ctor
order by dept_name desc, salary asc
```

Write a SQL statement to find the names of players that start with ‘S’ but does not end with ‘n’. (Database: FLIS)

```
select name as player_name from players
where name like 'S%'
except
select name as player_name from players
where name like '%n'
```

Find out the total number of players who are playing from the team id ‘T0001’. (Database: FLIS)

```
select team_id,count(team_id) as no_of_player from players
where team_id = 'T0001'
group by team_id
```

Write a query to obtain the natural join between the tables, students and departments.

(Database: LIS)

```
select *
from students natural join departments
```

Find details of those instructors of the Accounting department who have more salary than at least one instructor of the Psychology department. (Database: University)

```
select * from instructor
where dept_name = 'Accounting'
and salary > any (select salary from instructor
              where dept_name = 'Psychology')
```

Write a SQL statement to find out the manager’s date of birth(dob) of the team for which “Shlok” plays. (Database: FLIS)

```
select m.dob from managers m
join teams t on m.team_id = t.team_id
join players p on t.team_id = p.team_id
where p.name = 'Shlok'
```

```
SELECT dob
FROM managers
WHERE team_id IN (SELECT team_id
FROM players
WHERE name = ‘Shlok’)
```

Write a SQL statement to find out match number(match num) played in the playground “Emirates Stadium”. (Database: FLIS)

```
select match_num from matches
where host_team_id in (select team_id from teams
where playground = 'Emirates Stadium')
```

Find the name, player id, date of birth and city of all players who played for team ‘Rainbow’. (Database: FLIS)

```
select p.player_id,p.name,p.dob,t.city,t.name from players p
join teams t on p.team_id = t.team_id
where t.name = 'Rainbow'
```

Find the name of the teams which belong to the same city as the team ‘Amigos’.

(Database: FLIS)

```
SELECT t2.name
FROM teams AS t1, teams AS t2
WHERE t1.name = 'Amigos'
AND t2.name <> 'Amigos'
AND t1.city = t2.city
```

Find the name of the department in which Gita Das is studying. (Database: LIS)

```
select d.department_name from departments d
join students s on s.department_code = d.department_code
where s.student_fname = 'Gita' and s.student_lname = 'Das'
```

Find the roll number of all male students, having their department building in ‘Block 2’. (Database: LIS)

```
select s.roll_no, d.department_building from students s
join departments d on s.department_code = d.department_code
where d.department_building = 'Block_2' and s.gender = 'M'
```

Find the first name, last name and the roll number of students having their department building in ‘Block 1’. (Database: LIS)

```
select s.roll_no,s.student_fname,s.student_lname, d.department_building from
students s
```

```
join departments d on s.department_code = d.department_code
where d.department_building = 'Block_1
```

Find out the details of the members who have not issued any books. (Database: LIS)

```
SELECT *
FROM members
WHERE NOT EXISTS (SELECT *
FROM book_issue
WHERE members.member_no = book_issue.member_no)
```

Find out the name of courses which have been taught in both Fall semester and Spring semester. (Database: University)

```
SELECT c.title
FROM course c
WHERE c.course_id IN (
   SELECT course_id
   FROM section
   WHERE semester = 'Fall'

   INTERSECT

   SELECT course_id
   FROM section
   WHERE semester = 'Spring'
);
```

Write a SQL statement to find out the number of students who have studied in each building from 2005 till 2008 (including 2005,2008). (Database: University)

```
SELECT c.title
FROM course c
WHERE c.course_id IN (
   SELECT course_id
```

```
    FROM section
    WHERE semester = 'Fall'

    INTERSECT

    SELECT course_id
    FROM section
    WHERE semester = 'Spring'
);
```

Write a SQL query to find the host team id and the guest team id of the matches played before MAY 15, 2020, in which the host team scored more than 2 goals. [FLIS]

```
select host_team_id, guest_team_id from matches
where match_date < '2020-05-15' and host_team_score > 2
```

Write a SQL query to find the name of the managers whose names contain at least six characters. [FLIS]

```
select name from managers
where name like '______%'
```

Write a SQL query to find the member type and member number of the female students who are from the department code ‘EE’. [LIS]

```
select m.member_type, m.member_no from members m
join students s on m.roll_no = s.roll_no
where s.department_code = 'EE' and s.gender = 'F'
```

Write a SQL query to find the name of referees who were the fourth referee for match numbers ‘M0001’, ‘M0003’, ‘M0005’. [FLIS]

```
select r.name from referees r
join match_referees m on r.referee_id = m.fourth_referee
```

```
where m.match_num in ('M0001','M0003','M0005')
```

Write a SQL query to find the title, date of issue (doi) of the books issued by the students whose first name starts with ‘S’ and from departments ‘Mechanical Engineering’ or ‘Computer Science’. [LIS]

```
select a.doi , b.title from book_issue a
join book_copies c on a.accession_no = c.accession_no
join book_catalogue b on c.ISBN_no = b.ISBN_no
join members m on a.member_no = m.member_no
join students s on m.roll_no = s.roll_no
join departments d on s.department_code = d.department_code
where s.student_fname like 'S%' and d.department_name in ('Mechanical Engineering','Computer Science')
```

Write a SQL statement to find out the dates when one or more copies of the book having the title “Learning with Python” has been issued. (Database: LIS)

```
select bi.doi from book_issue bi
join book_copies bc on bi.accession_no = bc.accession_no
join book_catalogue bk on bc.ISBN_no = bk.ISBN_no
where bk.title = 'Learning with Python';
```

Write a SQL statement to find the name and salary of the instructor(s) having the max salary within that department whose avg salary is more than the max salary of Psychology dept. (Database: University)

```
SELECT i.name, i.salary FROM instructor i
WHERE i.salary = (
    SELECT MAX(i2.salary)
    FROM instructor i2
    WHERE i2.dept_name = i.dept_name
)
AND i.dept_name IN (
    SELECT dept_name
    FROM instructor
    GROUP BY dept_name
    HAVING AVG(salary) > (
        SELECT MAX(salary)
        FROM instructor
        WHERE dept_name = 'Psychology'
    )
);
```

Write a SQL statement to find out the number of courses which have been taught in Fall semester but never in Spring semester. (Database: University)

```
SELECT i.name, i.salary FROM instructor i
WHERE i.salary = (
    SELECT MAX(i2.salary)
    FROM instructor i2
    WHERE i2.dept_name = i.dept_name
)
AND i.dept_name IN (
    SELECT dept_name
    FROM instructor
    GROUP BY dept_name
    HAVING AVG(salary) > (
        SELECT MAX(salary)
        FROM instructor
        WHERE dept_name = 'Psychology'
    )
);
```

Print name,id,num count of instructor(s) who has taught maximum number of classes on the day ’W’. (num count is the number of the classes they took on day ’W’.) (Database: University)

```
SELECT i.ID, i.name, COUNT(*) AS num_count
FROM instructor i
JOIN teaches t ON i.ID = t.ID
JOIN section s ON t.course_id = s.course_id
```

```
       AND t.sec_id = s.sec_id
       AND t.semester = s.semester
       AND t.year = s.year
JOIN time_slot ts ON s.time_slot_id = ts.time_slot_id
WHERE ts.day = 'W'
GROUP BY i.ID, i.name
HAVING COUNT(*) = (
  SELECT MAX(class_count)
  FROM (
    SELECT COUNT(*) AS class_count
    FROM instructor i2
    JOIN teaches t2 ON i2.ID = t2.ID
    JOIN section s2 ON t2.course_id = s2.course_id
            AND t2.sec_id = s2.sec_id
            AND t2.semester = s2.semester
            AND t2.year = s2.year
    JOIN time_slot ts2 ON s2.time_slot_id = ts2.time_slot_id
    WHERE ts2.day = 'W'
    GROUP BY i2.ID
  ) AS sub
);
```

Find the names of the fourth referees (fourth referee) and the match number of all matches played on “2020-05-19”. (Database: FLIS)

```
SELECT m.match_num, r.name AS fourth_referee_name
FROM matches m
JOIN match_referees mr ON m.match_num = mr.match_num
JOIN referees r ON mr.fourth_referee = r.referee_id
WHERE m.match_date = '2020-05-19';
```

Write a SQL statement to find out the name of the oldest player from the team named “All Stars”. (Database: FLIS)

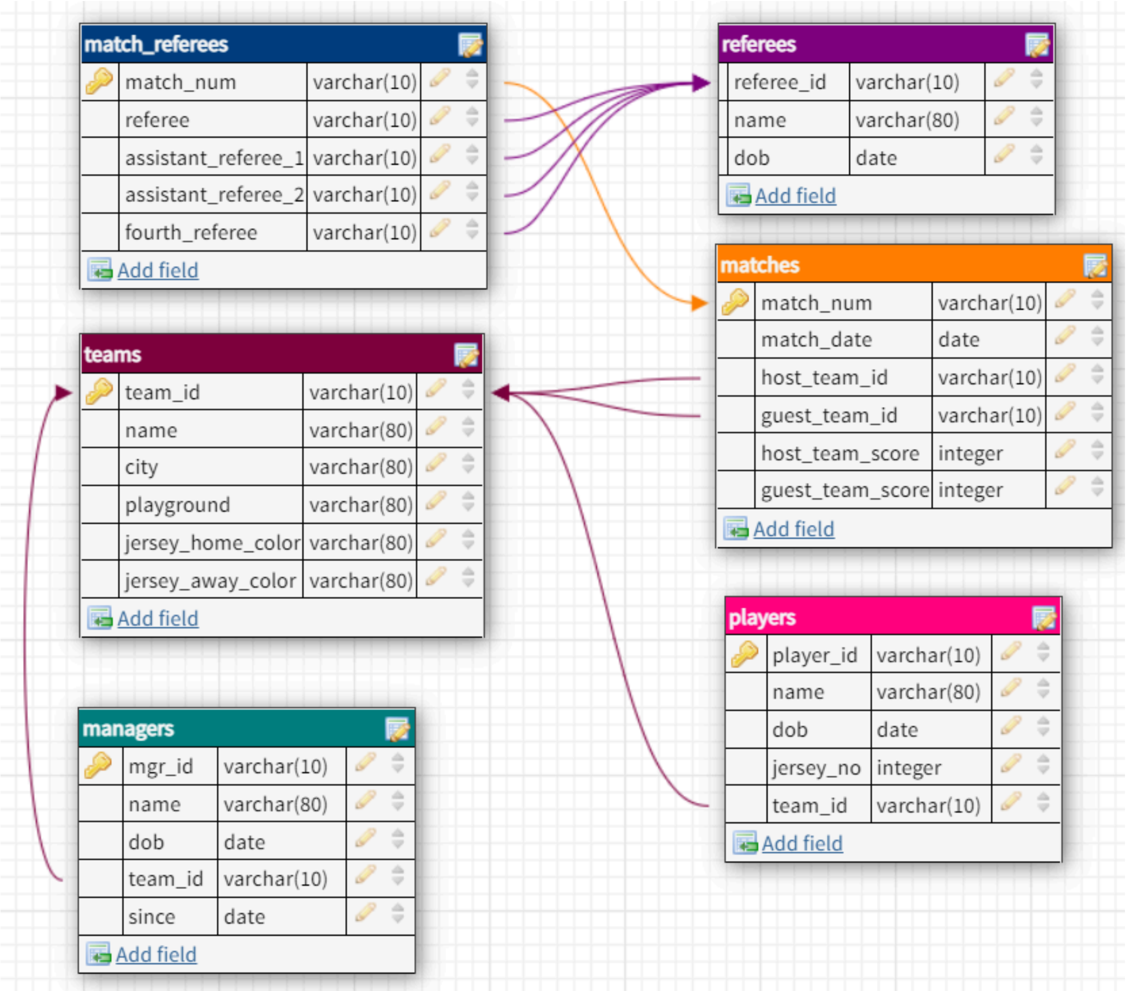
```
SELECT p.name
FROM players p
```

```
JOIN teams t ON p.team_id = t.team_id
WHERE t.name = 'All Stars'
  AND p.dob = (
      SELECT MIN(p2.dob)
      FROM players p2
      JOIN teams t2 ON p2.team_id = t2.team_id
      WHERE t2.name = 'All Stars'
  );
```

In this question, you must write a Python program to output the name of the main referee for a given match date (in yyyy-mm-dd format). The input to your program is a file named “date.txt” that has the match date as the first word of the file. Your program must assume that date.txt resides in the same folder as your Python program.

The output name has to be formatted as follows. The last name is displayed followed by the initials of the first name, then a full stop, a space and then the initials of the middle name (if the middle name exists), followed by a full stop.

- For example, if the name of the main referee is “Kennedy Sapam”, the output must be ”Sapam K.”
- If the name of the main referee is “Asit Kumar Sarkar”, the output must be ”Sarkar A. K.”



match_referees

match_num	varchar(10)
referee	varchar(10)
assistant_referee_1	varchar(10)
assistant_referee_2	varchar(10)
fourth_referee	varchar(10)

referees

referee_id	varchar(10)
name	varchar(80)
dob	date

teams

team_id	varchar(10)
name	varchar(80)
city	varchar(80)
playground	varchar(80)
jersey_home_color	varchar(80)
jersey_away_color	varchar(80)

matches

match_num	varchar(10)
match_date	date
host_team_id	varchar(10)
guest_team_id	varchar(10)
host_team_score	integer
guest_team_score	integer

players

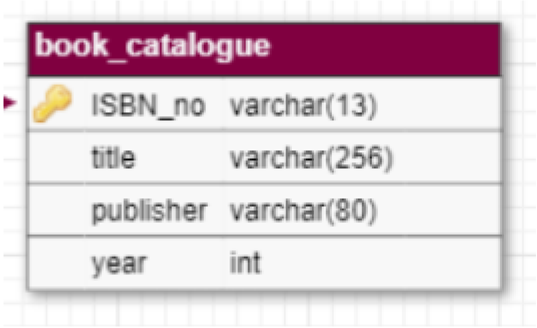
player_id	varchar(10)
name	varchar(80)
dob	date
jersey_no	integer
team_id	varchar(10)

managers

mgr_id	varchar(10)
name	varchar(80)
dob	date
team_id	varchar(10)
since	date

```
import psycopg2
import os
import sys

file = open('data.txt', 'r')
data = file.read()

try:
    connection = psycopg2.connect(
                                  database = sys.argv[1],
                                  user = os.environ.get('PGUSER'),
                                  password = os.environ.get('PGPASSWORD'),
                                  host = os.environ.get('PGHOST'),
                                port = os.environ.get('PGPORT'))

    cursor = connection.cursor()
    query = "select referees.name from match_refrees, matches, refrees
    where match_referees .match_num = matches.match_num and matches.ma
tch_date = '{}'
    and match_refrees.referee = referees.referee_id ".format(date)
    cursor.execute (query)

    result_of_query = cursor.fetchall()

    for result in result_of_query:
        list_of_name = list(map(str, result[0].split(" ")))
        final _result = list_of_name[-1]
        for name in list_of_name[:-1]:
                final_result += " "+name[0]"."
        print (final_result)
        cursor.close()

except(exception, psycopg2.DatabaseError) as error:
    print(error)
finally:
    connection.close
```

Note: Do not hard code the database name in your program, because your program will be run against a different database instance for evaluation. For the database connection, use the following connection string variables:database = sys.argv[1]   //name of the database is obtained from the command line argumentuser = os.environ.get('PGUSER') password = os.environ.get('PGPASSWORD') host = os.environ.get('PGHOST')port = os.environ.get('PGPORT')**Problem Statement:**



book_catalogue	
ISBN_no	varchar(13)
title	varchar(256)
publisher	varchar(80)
year	int

Write a Python program to print the ISBN numbers of books which are published in a given year. Here, the year is obtained as the value of function L(x) (given after the sample output) at x. You have to read the value of x from the input file "**number.txt**", and use it to find the value of L(x). Your program must assume that the file **number.txt** resides in the same folder as your Python program.
You have to iterate through the list and print each value separately as shown in the output below:

```
9789352921171
9789351343202
9789353333380
```

The line function is given below:

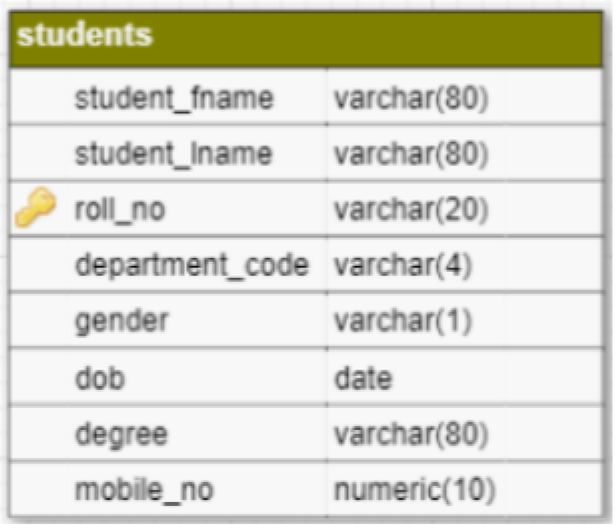
L4(x) = 2000 + 4*x + 4

```
import psycopg2
import sys
import os

database = sys.argv[1]
user = os.environ.get('PGUSER')
password = os.environ.get('PGPASSWORD')
host = os.environ.get('PGHOST')
port = os.environ.get('PGPORT')

with open("number.txt", "r") as f:
    x = int(f.read().strip())

year = 2000 + 4 * x + 4
```

```
try:
    connection = psycopg2.connect(
        dbname=database,
        user=user,
        password=password,
        host=host,
        port=port
    )
    cursor = connection.cursor()

    query = "SELECT ISBN_no FROM book_catalogue WHERE year = %s;"
    cursor.execute(query, (year,))

    results = cursor.fetchall()
    for row in results:
        print(row[0])

except Exception as e:
    print("Error:", e)

finally:
    if cursor:
        cursor.close()
    if connection:
        connection.close()
```

Note: Do not hard code the database name in your program, because your program will be run against a different database instance for evaluation. For the database connection, use the following connection string variables:database = sys.argv[1]   //name of the database is obtained from the command line argumentuser = os.environ.get('PGUSER') password = os.environ.get('PGPASSWORD') host = os.environ.get('PGHOST')port = os.environ.get('PGPORT')

Problem Statement:



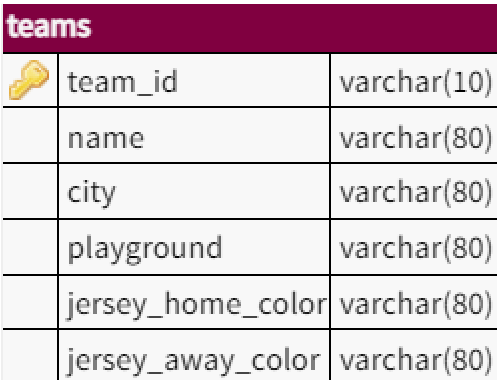
students		
	student_fname	varchar(80)
	student_lname	varchar(80)
🔑	roll_no	varchar(20)
	department_code	varchar(4)
	gender	varchar(1)
	dob	date
	degree	varchar(80)
	mobile_no	numeric(10)

Write a Python program to print the roll number of the student. Student's first name is given in a file named 'name.txt' resides in the same folder as python program file. The output of the python program is only roll number. For example, if the first name of the student is 'Vikas'. Then output must be CS01 only. Note: No spaces.

```
import psycopg2
import os, sys

f = open('name.txt','r')
idx = f.read()

try:
    connection = psycopg2.connect(database = sys.argv[1],
    user = os.environ.get('PGUSER'),
    password = os.environ.get('PGPASSWORD'),
    host = os.environ.get('PGHOST'),
    port = os.environ.get('PGPORT'))

    cursor = connection.cursor()
    query = f"select roll_no from students where student_fname = '{idx}'"
    cursor.execute(query)

    result = cursor.fetchall()
    for i in result:
        print(i[0])

    cursor(close)

except(Exception, psycopg2.DatabaseError) as Error:
    print(error)

finally:
    connection.close()
```

Note: Do not hard code the database name in your program, because your program will be run against a different database instance for evaluation. For the database connection, use the following connection string variables:database = sys.argv[1]   //name of the database is obtained from the command line argumentuser = os.environ.get('PGUSER') password = os.environ.get('PGPASSWORD') host = os.environ.get('PGHOST')port = os.environ.get('PGPORT')

Problem Statement:



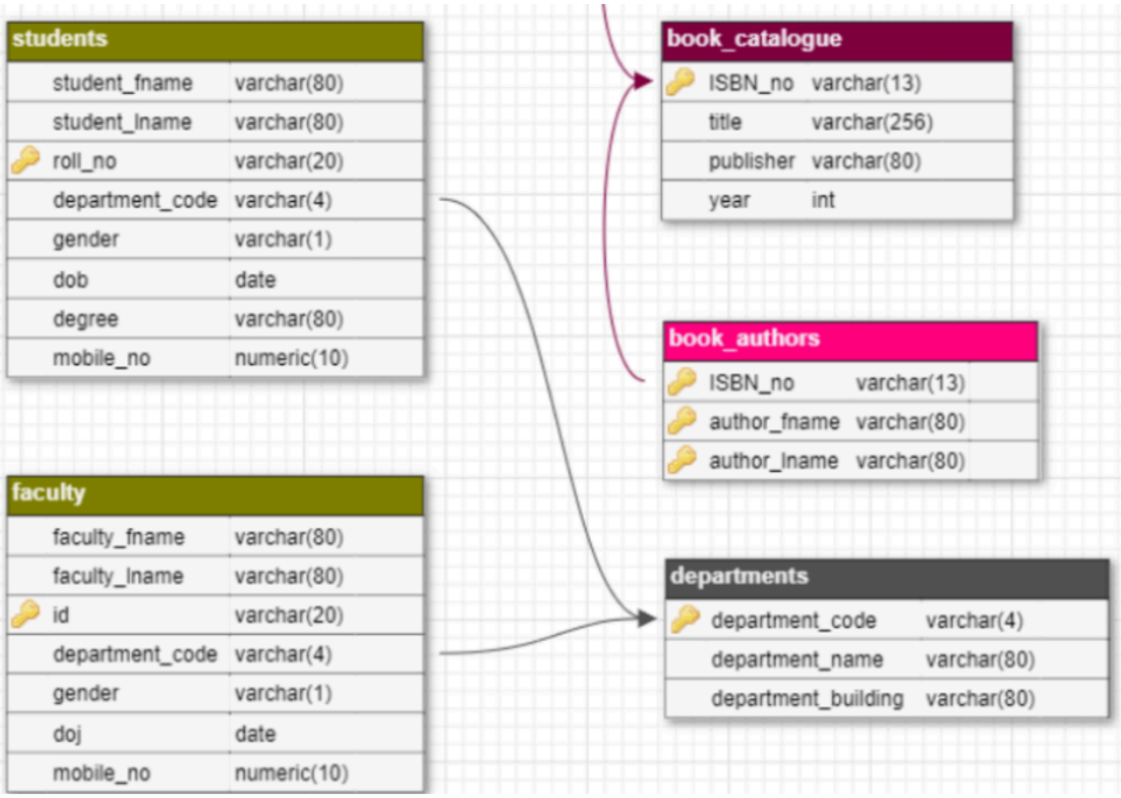
teams		
🔑	team_id	varchar(10)
	name	varchar(80)
	city	varchar(80)
	playground	varchar(80)
	jersey_home_color	varchar(80)
	jersey_away_color	varchar(80)

write a Python program to print the playground of the given team id. team_id is given in a file named 'team.txt' resides in the same folder as python program file. • The output of the python program is only playground name. • For example, if the team_id is 'T0002' . Then output must be **Villa Park** only

```
import psycopg2
import os
import sys

f = open('team.txt','r')
idx = f.read()

try:
    connection = psycopg2.connect(database = sys.argv[1],
    user = os.environ.get('PGUSER'),
    password = os.environ.get('PGPASSWORD'),
    host = os.environ.get('PGHOST'),
    port = os.environ.get('PGPORT'))

    cursor = connection.cursor()
    query = f"select playground from teams where team_id = '{idx}' "
    cursor.execute(query)

    result = cursor.fetchall()
    for i in result:
        print(i[0])

    cursor.close()

except(Exception, psycopg2.DatabaseError) as Error:
    print(error)



finally:
    connection.close()
```

Note: Do not hard code the database name in your program, because your program will be run against a different database instance for evaluation. For the database connection, use the following connection string variables:database = sys.argv[1]   //name of the database is obtained from the command line argumentuser = os.environ.get('PGUSER') password = os.environ.get('PGPASSWORD') host = os.environ.get('PGHOST')port = os.environ.get('PGPORT')

Problem Statement:



students	
student_fname	varchar(80)
student_lname	varchar(80)
🔑 roll_no	varchar(20)
department_code	varchar(4)
gender	varchar(1)
dob	date
degree	varchar(80)
mobile_no	numeric(10)

faculty	
faculty_fname	varchar(80)
faculty_lname	varchar(80)
🔑 id	varchar(20)
department_code	varchar(4)
gender	varchar(1)
doj	date
mobile_no	numeric(10)

book_catalogue	
🔑 ISBN_no	varchar(13)
title	varchar(256)
publisher	varchar(80)
year	int

book_authors	
🔑 ISBN_no	varchar(13)
🔑 author_fname	varchar(80)
🔑 author_lname	varchar(80)

departments	
🔑 department_code	varchar(4)
department_name	varchar(80)
department_building	varchar(80)

Write a Python program to print the student's first name, the corresponding department name and the respective year of date of birth, if the year is even then print "Even" or else "Odd".Student's first name is given in a file named 'name.txt' resides in the same folder as python program file.The output of the python program is only student's first name, the corresponding department name and year of date of birth, if the year is even then "Even" or else "Odd".For example, 'Suman' and 'Computer Science' is the name and department name of the student. '2002' is the year he was born in. '2002' is even. Then, the final output will be **Suman,Computer Science,Even** only. Note: No spaces.For example, 'Vinod' and 'Electrical Engineering' is the name and department name of the student. '2003' is the year he was born in. '2003' is not even. Then, the final output will be **Vinod,Electrical Engineering,Odd** only. Note: No spaces.

- The output of the python program is only student's first name, the corresponding department name and year of date of birth, if the year is even then "Even" or else "Odd".
- For example, 'Suman' and 'Computer Science' is the name and department name of the student. '2002' is the year he was born in. '2002' is even. Then, the final output will be **Suman,Computer Science,Even** only. Note: No spaces.
- For example, 'Vinod' and 'Electrical Engineering' is the name and department name of the student. '2003' is the year he was born in. '2003' is not even. Then, the final output will be **Vinod,Electrical Engineering,Odd** only. Note: No spaces.

```
import psycopg2
import sys
import os

# Database connection parameters
database = sys.argv[1]  # database name from command line
user = os.environ.get('PGUSER')
password = os.environ.get('PGPASSWORD')
host = os.environ.get('PGHOST')
port = os.environ.get('PGPORT')

# Step 1: Read student first name from file
with open("name.txt", "r") as f:
    student_fname = f.read().strip()

try:
    # Step 2: Connect to PostgreSQL
    connection = psycopg2.connect(
        dbname=database,
        user=user,
        password=password,
        host=host,
        port=port
    )
    cursor = connection.cursor()

    # Step 3: Query to get student's info
    query = """
        SELECT s.student_fname, d.department_name, EXTRACT(YEAR FROM s.dob) AS birth_year
        FROM students s
        JOIN departments d ON s.department_code = d.department_code
        WHERE s.student_fname = %s;
    """
    cursor.execute(query, (student_fname,))
    result = cursor.fetchone()

    if result:
        fname, dept, year = result
        year = int(year)
        parity = "Even" if year % 2 == 0 else "Odd"
        print(f"{fname},{dept},{parity}")
    else:
        print("No student found")

except Exception as e:
    print("Error:", e)

finally:
    if cursor:
        cursor.close()
    if connection:
        connection.close()
```