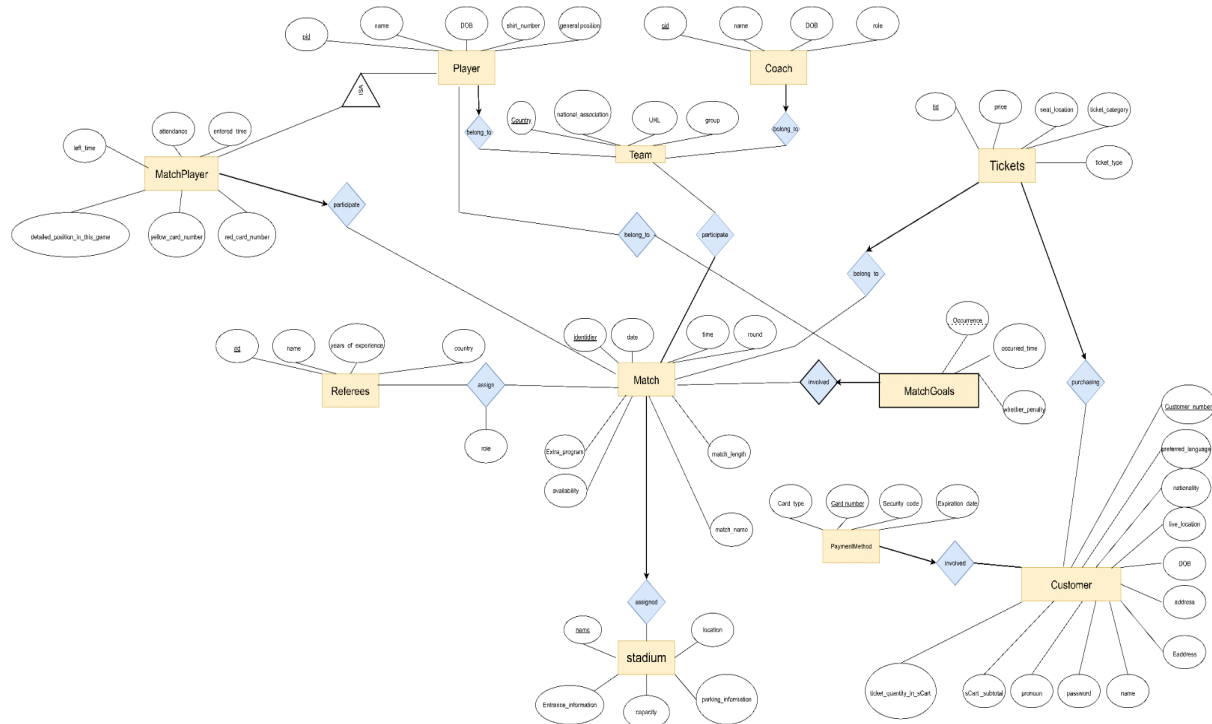


## Relational Schema



## Pending constraints

1. A match referee group needs 4 common referees and 4 VAR referees. We can't guarantee there are two groups with eight referees.
2. We can't guarantee a maximum number of tickets for each match.
3. It is impossible to guarantee there will be exactly 4 coaches for each team since it is a one to many relation (Every coach belongs to exactly one team but it is possible that one team has more than one coach).
4. We cannot dictate how many tickets a consumer can buy.
5. We cannot specify the upper limit of the players and coaches a team could have.
6. We cannot guarantee there are exactly 11 players playing for each team when the game starts.

5.

a. Write a SQL query that lists all the stadium names and their locations and the match date of matches in which player Christine Sinclair has played and scored at least one goal. You can assume that there is only one player with this particular name (and you can change the actual player name if you have a particular favorite).

```

C94218216> SELECT s.name, s.location, m.date
FROM STADIUM s INNER JOIN MATCH m
ON s.name = m.stadium
WHERE m.identidier IN ( SELECT identidier
FROM Player p INNER JOIN MatchGoals
ON p.pid = MatchGoals.who_scored
WHERE p.name = 'Kyllian Mbappe' AND MATCHGoals.occurrence >=1)
AND m.identidier IN ( SELECT ATTENDANCE
FROM Match_Player
WHERE MATCH_PLAYER.ATTENDANCE = true)
[2023-02-26 14:51:16] 2 rows retrieved starting from 1 in 116 ms (execution: 73 ms, fetching: 43 ms)

```

The screenshot shows a database IDE with a SQL query editor and a results pane. The query is as follows:

```

1  --a. Write a SQL query that lists all the stadium names and their locations and
2  --in which player Christine Sinclair has played and scored at least one goal.
3  -- I will change Christine Sinclair into Kylian Mbappe, whose pid = 9000012
4  SELECT s.name, s.location, m.date
5  FROM STADIUM s INNER JOIN MATCH m
6       ON s.name = m.stadium
7  WHERE m.identidier IN ( SELECT identidier
8                          FROM Player p INNER JOIN MatchGoals
9                          ON p.pid = MatchGoals.who_scored
10                         WHERE p.name = 'Kylian Mbappe' AND MatchGoals.occurrence >=1)
11 AND m.identidier IN ( SELECT ATTENDANCE
12                      FROM Match_Player
13                      WHERE MATCH_PLAYER.ATTENDANCE = true);
14

```

The results pane shows the following data:

NAME	LOCATION	DATE
Al-Rayyan Stadium	Al Wakrah, Qatar	2023-06-01
Lusail Iconic Stadium	Lusail, Qatar	2023-06-15

**b. Write a SQL query that lists the name, shirt number and country of all players that have played in all matches of their teams.**

```

CS421G216> SELECT name, shirt_number, country
            FROM PLAYER
            WHERE PLAYER.pid IN (SELECT pid
                                FROM MATCH_PLAYER
                                WHERE MATCH_PLAYER.attendance = true
                                EXCEPT
                                SELECT pid
                                FROM MATCH_PLAYER
                                WHERE MATCH_PLAYER.attendance = false)

[2023-02-26 14:53:48] 8 rows retrieved starting from 1 in 70 ms (execution: 21 ms, fetching: 49 ms)

```

```

19 SELECT name,shirt_number,country
20 FROM PLAYER
21 WHERE PLAYER.pid IN (SELECT pid
22                       FROM MATCH_PLAYER
23                       WHERE MATCH_PLAYER.attendance = true
24                       EXCEPT
25                       SELECT pid
26                       FROM MATCH_PLAYER
27                       WHERE MATCH_PLAYER.attendance = false);
  
```

Output: CS421G216.PLAYER

	NAME	SHIRT_NUMBER	COUNTRY
1	Lionel Messi	10	Argentina
2	Sergio Agüero	19	Argentina
3	Giovani Lo Celso	20	Argentina
4	Casemiro	5	Brazil
5	Manuel Neuer	1	Germany
6	Serge Gnabry	20	Germany
7	Kylian Mbappe	7	France
8	Antoine Griezmann	10	France

c. Write a SQL query that lists for each team, the country, the number of matches they have played and the total number of goals they have scored during normal play.

the result:

```

32 with goal_info(pid,goals) AS
33 (
34     SELECT who_scored,COUNT(who_scored) as num_goals
35     FROM Matchgoals m
36     GROUP BY who_scored
37 )
38 SELECT Country,COUNT(goals) AS goal_num
39 FROM Player INNER JOIN goal_info
40 ON Player.PID = goal_info.pid
41 GROUP BY Country;
  
```

Output: Result 42

	COUNTRY	COUNT_TOTAL	GOAL_NUM
1	Argentina	3	1
2	Brazil	5	0
3	England	3	2
4	France	5	2
5	Germany	4	1
6	Spain	4	0

and the code has been executed successfully

```
CS421G216> with total_info(country,count_total) AS
      (SELECT Match.country, SUM(Match.COUNTS) AS COUNT_TOTAL
        FROM (SELECT Match.h_name AS country, COUNT(*) AS COUNTS
              FROM Match
              GROUP BY h_name
              UNION ALL
              SELECT Match.v_name AS country, COUNT(*) AS COUNTS
              FROM Match
              GROUP BY v_name) Match
        GROUP BY Match.country)
      SELECT TEAM.country,count_total,goal_num
      FROM total_info INNER JOIN TEAM
      ON TEAM.COUNTRY = total_info.country
[2023-02-26 23:37:45] 6 rows retrieved starting from 1 in 102 ms (execution: 39 ms, fetching: 63 ms)
```

I add a attribute,number of goals , in the table Team

SELECT \*

FROM TEAM;

	COUNTRY	NATIONAL_ASSOCIATION	URL	GROUP	GOAL_NUM
1	Argentina	Argentine Football Association	http://www.afa.org.ar/	A	1
2	Brazil	Brazilian Football Confederation	https://www.cbf.com.br/	G	0
3	Germany	German Football Association	https://www.dfb.de/	E	1
4	England	The Football Association	http://www.thefa.com/	B	2
5	Spain	Royal Spanish Football Federation	https://www.rfef.es/	E	0
6	France	French Football Federation	https://www.fff.fr/	D	2

by using the query

The screenshot shows a SQL IDE interface. The main editor displays a SQL query with a CTE and a main SELECT statement. The query is as follows:

```

32 with goal_info(pid,goals) AS
33 (
34     SELECT who_scored,COUNT(who_scored) as num_goals
35     FROM Matchgoals m
36     GROUP BY who_scored
37 )
38 SELECT Country,COUNT(goals) AS goal_num
39 FROM Player INNER JOIN goal_info
40 ON Player.PID = goal_info.pid
41 GROUP BY Country;
42

```

The right sidebar shows a database schema for 'project2\_DB2' with tables: COACH, CUSTOMER, MATCH, MATCH\_PLAYER, MATCHGOALS, PLAYER, REFEREES, REFEREES\_ASSIGN, and STADIUM.

The bottom panel shows the 'Output' window with 'Result 47' containing 4 rows of data:

COUNTRY	GOAL_NUM
1 Argentina	1
2 England	2
3 France	2
4 Germany	1

d.Create an interesting SQL query that extracts some information from tables that refers to purchasing tickets, e.g., some summary information about tickets sold for a particular match, information how many tickets were sold for a match / each match and whether the stadium was sold out, or anything else that might be interesting. The query should not only be a simple query on a single table with only basic selections and projections.

```

1  ✓ select MATCH.IDENTIDIER,MATCH_NAME,PRICE * AVAILABILITY as Profit,TICKET_TYPE
2      from TICKETS join MATCH
3      on MATCH.IDENTIDIER = TICKETS.IDENTIDIER
4  order by Profit;

```

Output Result 25

6 rows

	IDENTIDIER	MATCH_NAME	PROFIT	TICKET_TYPE
1	10001	France vs Argentina	475000	Child
2	10002	England vs Brazil	600000	Child
3	10001	France vs Argentina	602500	Adult
4	10001	France vs Argentina	1000000	VIP
5	10002	England vs Brazil	1125000	Adult
6	10002	England vs Brazil	1875000	VIP

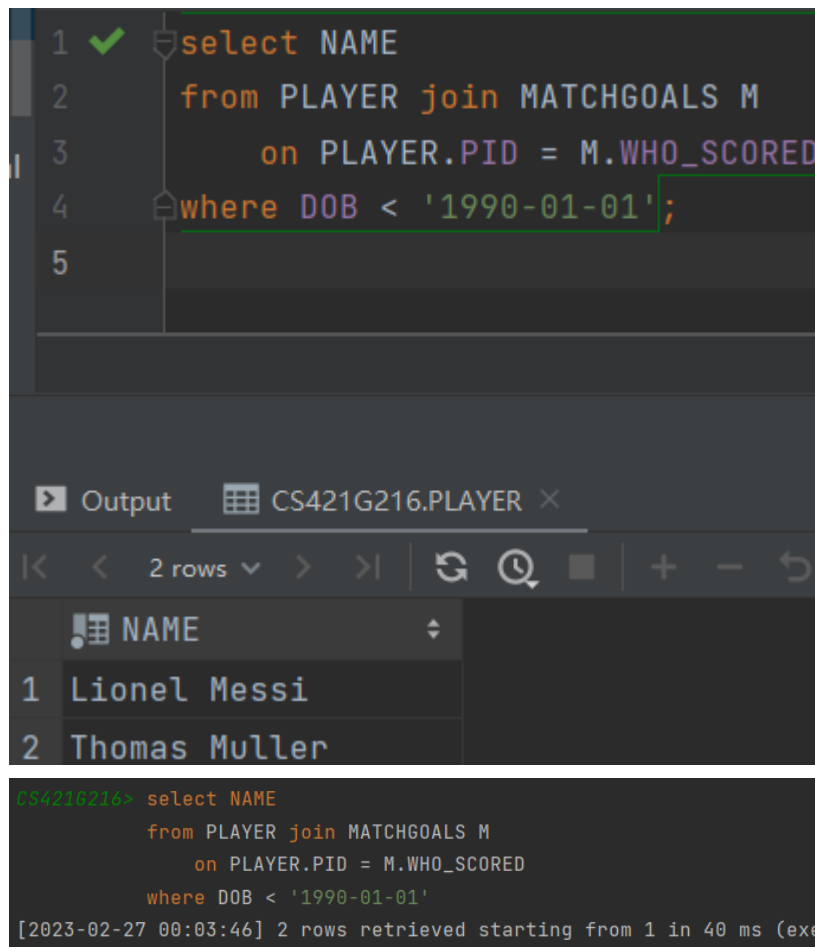
```

PS4210216> select MATCH.IDENTIDIER,MATCH_NAME,PRICE * AVAILABILITY as Profit,TICKET_TYPE
              from TICKETS join MATCH
              on MATCH.IDENTIDIER = TICKETS.IDENTIDIER
              order by Profit
[2023-02-27 00:04:59] 6 rows retrieved starting from 1 in 42 ms (execution: 15 ms, fetching: 27 ms)

```

As shown above, we can know the profits for each type of ticket(child,adult,vip) for each game.

**e. Create a further SQL query that is of interest for this soccer world cup database. Maybe it uses some tables that are not used in any of the other queries, or performs some conditions on the date/time attributes of the schema or any other attributes that have not been used in one of the other queries.**



The screenshot shows a SQL IDE interface. The top pane contains a SQL query: `select NAME from PLAYER join MATCHGOALS M on PLAYER.PID = M.WHO_SCORED where DOB < '1990-01-01';`. The bottom pane shows the output of the query, which is a table with two rows: `1 Lionel Messi` and `2 Thomas Muller`. Below the output, there is a terminal window showing the same query and its execution details: `[2023-02-27 00:03:46] 2 rows retrieved starting from 1 in 40 ms (execution: 15 ms, fetching: 25 ms)`.

```
1 ✓ select NAME
2   from PLAYER join MATCHGOALS M
3     on PLAYER.PID = M.WHO_SCORED
4   where DOB < '1990-01-01';
5
```

Output CS421G216.PLAYER ×

	NAME
1	Lionel Messi
2	Thomas Muller

```
CS421G216> select NAME
              from PLAYER join MATCHGOALS M
                on PLAYER.PID = M.WHO_SCORED
              where DOB < '1990-01-01'
[2023-02-27 00:03:46] 2 rows retrieved starting from 1 in 40 ms (execution: 15 ms, fetching: 25 ms)
```

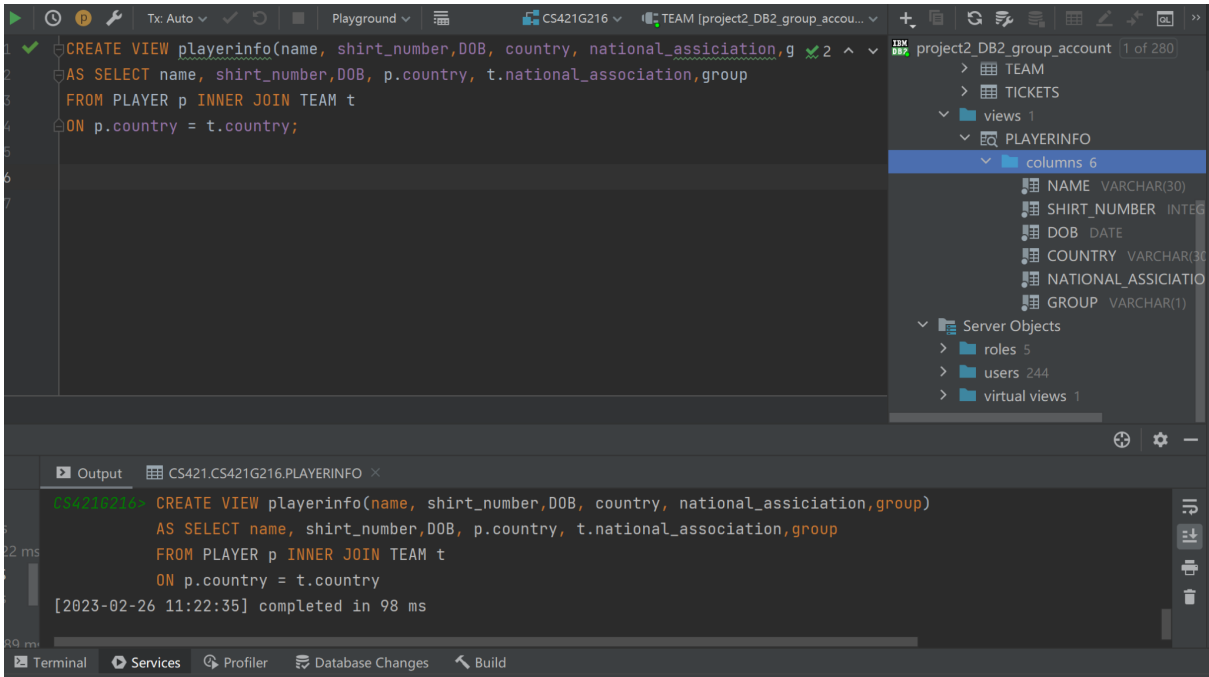
Which “old” player still carries the team? By comparing their ages and whether they have scored we can get such players.

6.

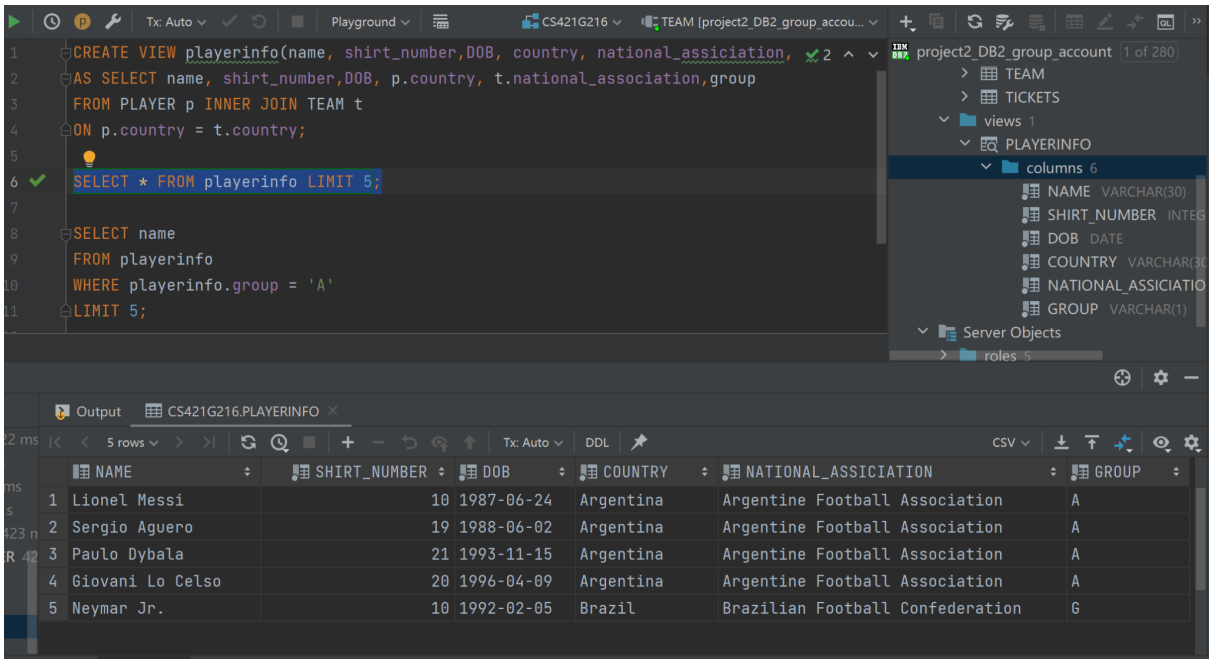
**a.Include the view definition SQL query (as plain text).**

A SQL view is a virtual table that is created from the output of a SQL statement and is similar to a real table in terms of the number of rows and columns it contains. The fields contained in a view correspond to fields in one or more actual tables in the database.

**b.Screenshot of the view creation being a success.**



c. Screenshot of a SQL query that selects everything from the view, truncated to just 5 records.





The screenshot shows a SQL IDE with a query execution log at the bottom. The log displays the following queries and their execution details:

```

MATCHGOALS 122 ms
5.a.sql 122 ms
createtbl 3 s 136 ms
MATCH_PLAYER 423 ms
TEAM 205 ms
6.sql 205 ms

```

The database schema view on the right shows the following structure:

- project2\_DB2\_group\_account 1 of 280
  - TEAM
  - TICKETS
  - views 1
    - PLAYERINFO
      - columns 6
        - NAME VARCHAR(30)
        - SHIRT\_NUMBER INTEG
        - DOB DATE
        - COUNTRY VARCHAR(30)
        - NATIONAL\_ASSOCIATIO
        - GROUP VARCHAR(1)
  - Server Objects
    - roles 5

d. Screenshot of a SQL query on the view that limits the previous output to only the players that belong to teams that belong to group "Group A". Truncate the output to just 5 records.

The screenshot shows a SQL IDE with a query execution log at the bottom. The log displays the following queries and their execution details:

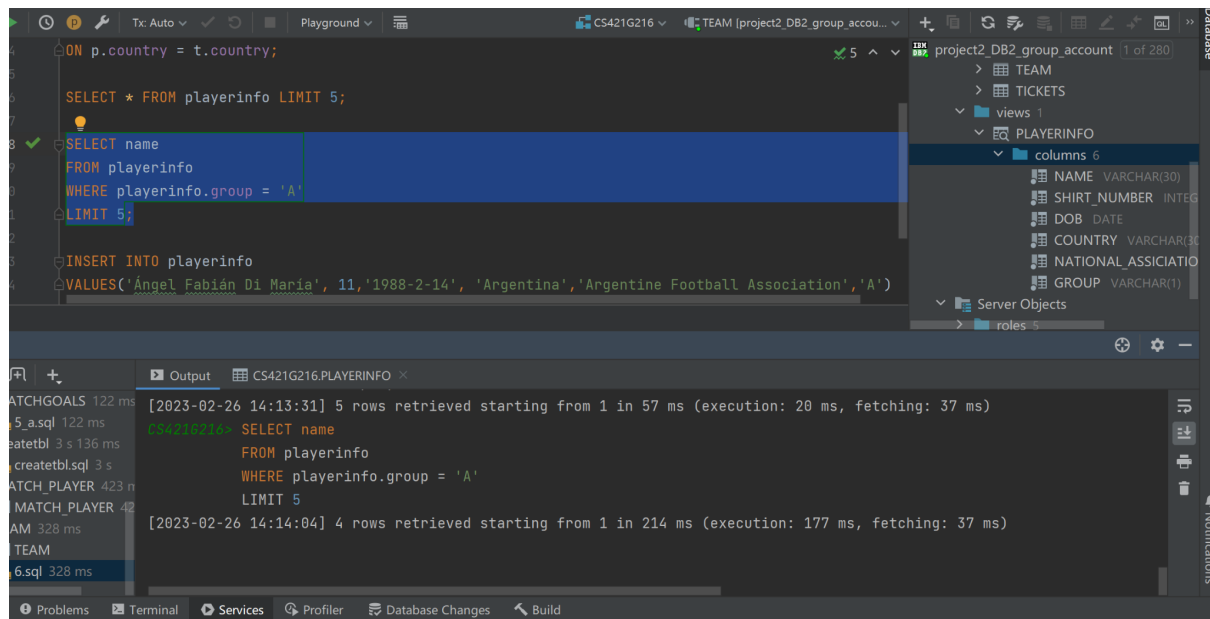
```

MATCHGOALS 122 ms
5.a.sql 122 ms
createtbl 3 s 136 ms
MATCH_PLAYER 423 ms
TEAM 205 ms
6.sql 205 ms

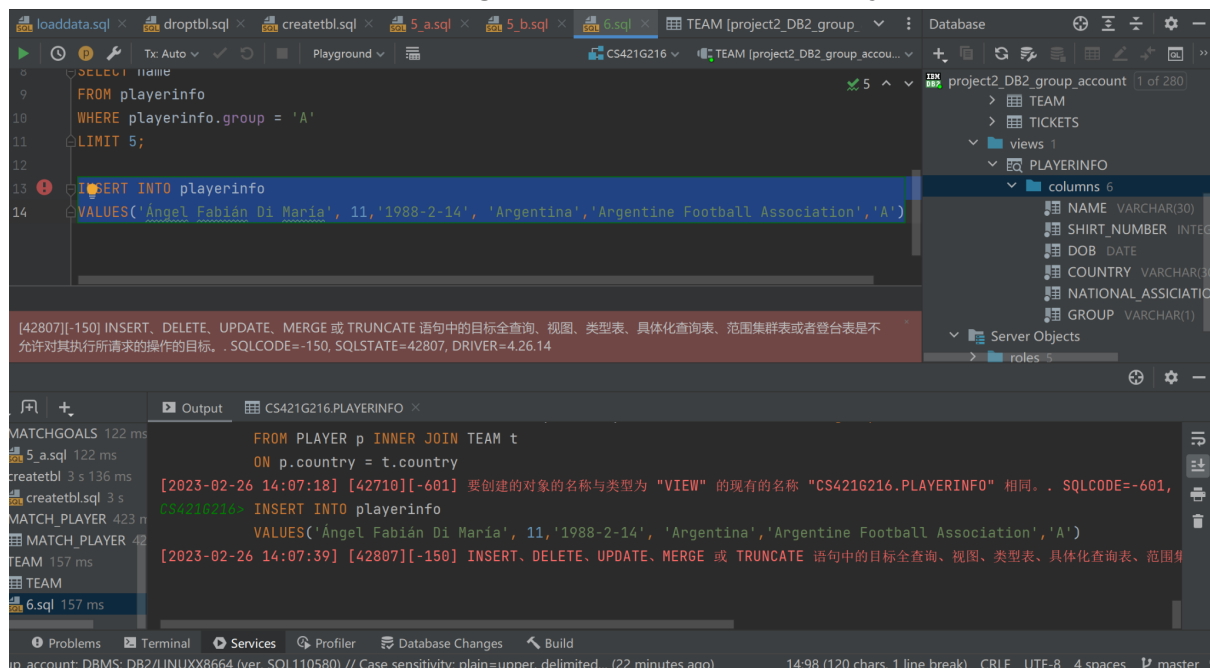
```

The database schema view on the right shows the following structure:

- project2\_DB2\_group\_account 1 of 280
  - TEAM
  - TICKETS
  - views 1
    - PLAYERINFO
      - columns 6
        - NAME VARCHAR(30)
        - SHIRT\_NUMBER INTEG
        - DOB DATE
        - COUNTRY VARCHAR(30)
        - NATIONAL\_ASSOCIATIO
        - GROUP VARCHAR(1)
  - Server Objects
    - roles 5

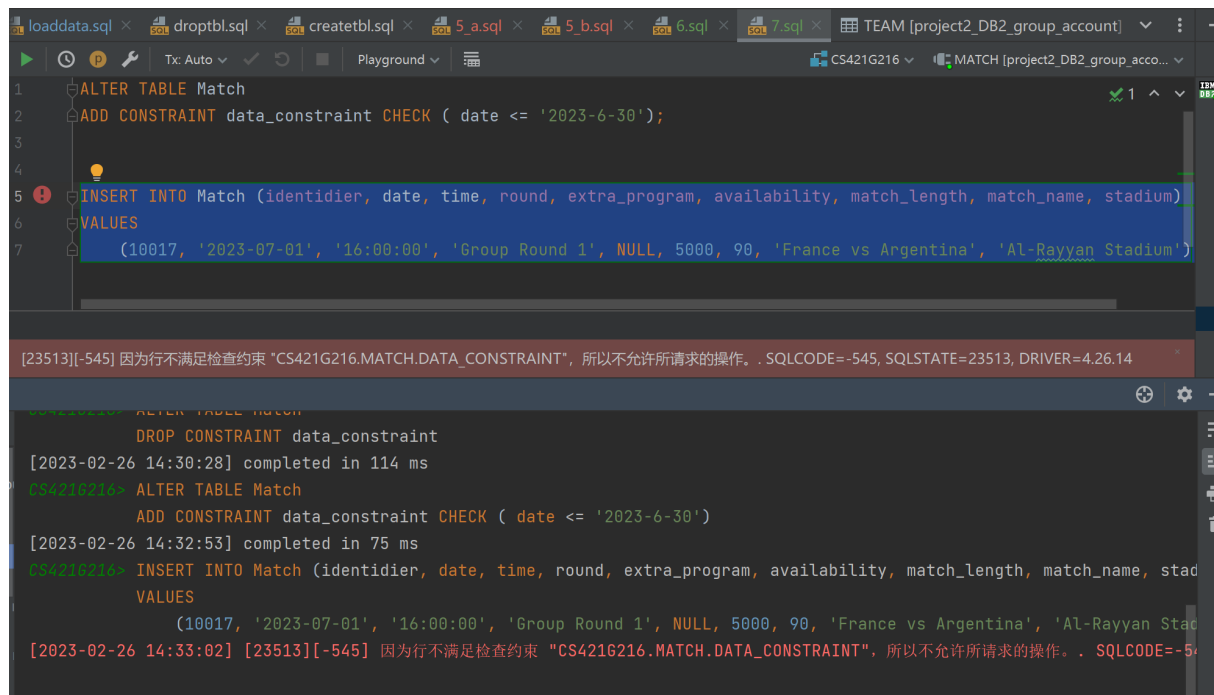


e. Now try inserting a record into the view (name, shirt number, DOB, country, etc., i.e., the attributes that are mentioned in the view's output) that has valid domain values for these attributes (e.g. a new name, but an existing group). Observe what happens. Take a screenshot and turn in that along with the explanation of why this happened.



The inserting value is not allowed. We cannot insert into view since view is not creating a table, it is an unmaterialized relation.

## 7. Check Constraints



The screenshot shows a SQL IDE with multiple tabs. The active tab is '7.sql'. The SQL editor contains the following code:

```
1 ALTER TABLE Match
2 ADD CONSTRAINT data_constraint CHECK ( date <= '2023-6-30');
3
4
5 INSERT INTO Match (identidier, date, time, round, extra_program, availability, match_length, match_name, stadium)
6 VALUES
7 (10017, '2023-07-01', '16:00:00', 'Group Round 1', NULL, 5000, 90, 'France vs Argentina', 'Al-Rayyan Stadium')
```

The execution results pane at the bottom shows the following output:

```
CS421G216> ALTER TABLE Match
          DROP CONSTRAINT data_constraint
[2023-02-26 14:30:28] completed in 114 ms
CS421G216> ALTER TABLE Match
          ADD CONSTRAINT data_constraint CHECK ( date <= '2023-6-30')
[2023-02-26 14:32:53] completed in 75 ms
CS421G216> INSERT INTO Match (identidier, date, time, round, extra_program, availability, match_length, match_name, stad
          VALUES
          (10017, '2023-07-01', '16:00:00', 'Group Round 1', NULL, 5000, 90, 'France vs Argentina', 'Al-Rayyan Stad
[2023-02-26 14:33:02] [23513][-545] 因为行不满足检查约束 "CS421G216.MATCH.DATA_CONSTRAINT", 所以不允许所请求的操作。 . SQLCODE=-545, SQLSTATE=23513, DRIVER=4.26.14
```

I constraint that the match data should be earlier than 2023-06-30.

I insert a record where the match data is 2023-07-01, which is later than 2023-06-30. So it is not allowed.