System Designs & Databases ICA

T-SQL SERVER – T-SQL QUERIES TO SUPPORT

European Top Leagues

Name: **Carl Baines**

Course: **System Designs and Databases**

Date**: 18/03/2025**

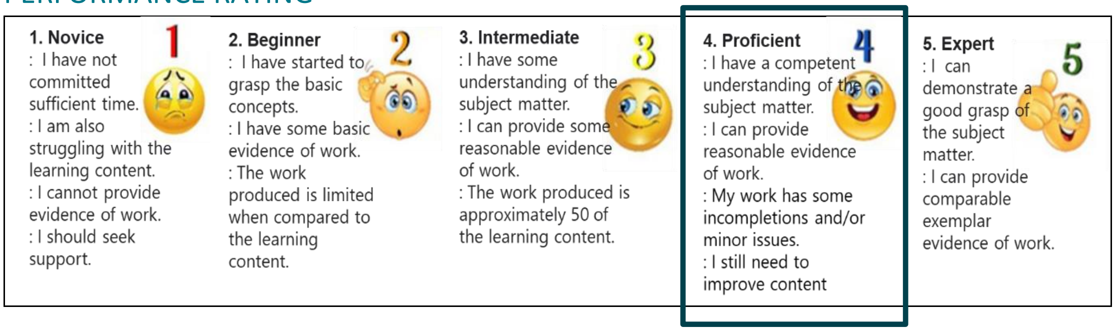
Tutor Name: **Sumeia Elkazza**

Table of Contents

T-SQL Server Practitioner Details

SQL Server - TSQL Practitioner Details:		
Name:	**Carl Baines**	
Email Address:	E4092399@live.tees.ac.uk	
Course:	BsC (Hons) Computer Science	
Date:	18/03/2025	
	Tutor:	Sumeia Elkazza

PERFORMANCE RATING



1. Novice : I have not committed sufficient time. : I am also struggling with the learning content. : I cannot provide evidence of work. : I should seek support.	**2. Beginner** : I have started to grasp the basic concepts. : I have some basic evidence of work. : The work produced is limited when compared to the learning content.	**3. Intermediate** : I have some understanding of the subject matter. : I can provide some reasonable evidence of work. : The work produced is approximately 50 of the learning content.	**4. Proficient** : I have a competent understanding of the subject matter. : I can provide reasonable evidence of work. : My work has some incompletions and/or minor issues. : I still need to improve content	**5. Expert** : I can demonstrate a good grasp of the subject matter. : I can provide comparable exemplar evidence of work.

INTRO

As a BsC (Hons) Computer Science student at university, I've developed a strong interest with software development, particularly in the areas of frontend design and backend databases. I decided upon studying at university to gather experience of using programming languages and data management tools like SQL, whilst also developing my problem-solving skills along the way. I am pursuing my interests as a graduate developer, as I want to demonstrate that I can contribute to projects that require both technical knowledge and creativity.

WHY YOU SHOULD LEARN T-SQL

I recommend that someone should learn SQL because it is the standard language for managing and manipulating databases. T-SQL, which is Transact-SQL, is Microsoft's version extension of SQL; it is a powerful tool worth learning for working with SQL server. Another reason why I recommend that someone should pick up SQL is because it is used predominantly in the industry, across many sectors. I hope to gain experience with writing complex queries and performing data analysis.

Hyperlinks to Graduate SQL Jobs

Jackson Hogg - SQL Developer
https://www.linkedin.com/jobs/search/?currentJobId=4215893507&f_C=2845536&geoId=9

2000000&origin=COMPANY_PAGE_JOBS_CLUSTER_EXPANSION&originToLandingJobPostings=4215893507%2C4209730541%2C4183352505%2C4214420070%2C4208018253%2C4205505738%2C4214406967%2C4209779049%2C4205571241&trk=d_flagship3_company

Recorra - Senior MS SQL/Access Developer

https://uk.indeed.com/jobs?q=sql&start=10&vjk=119ad36ef0b6259b

Avanade – SQL Database Administrator

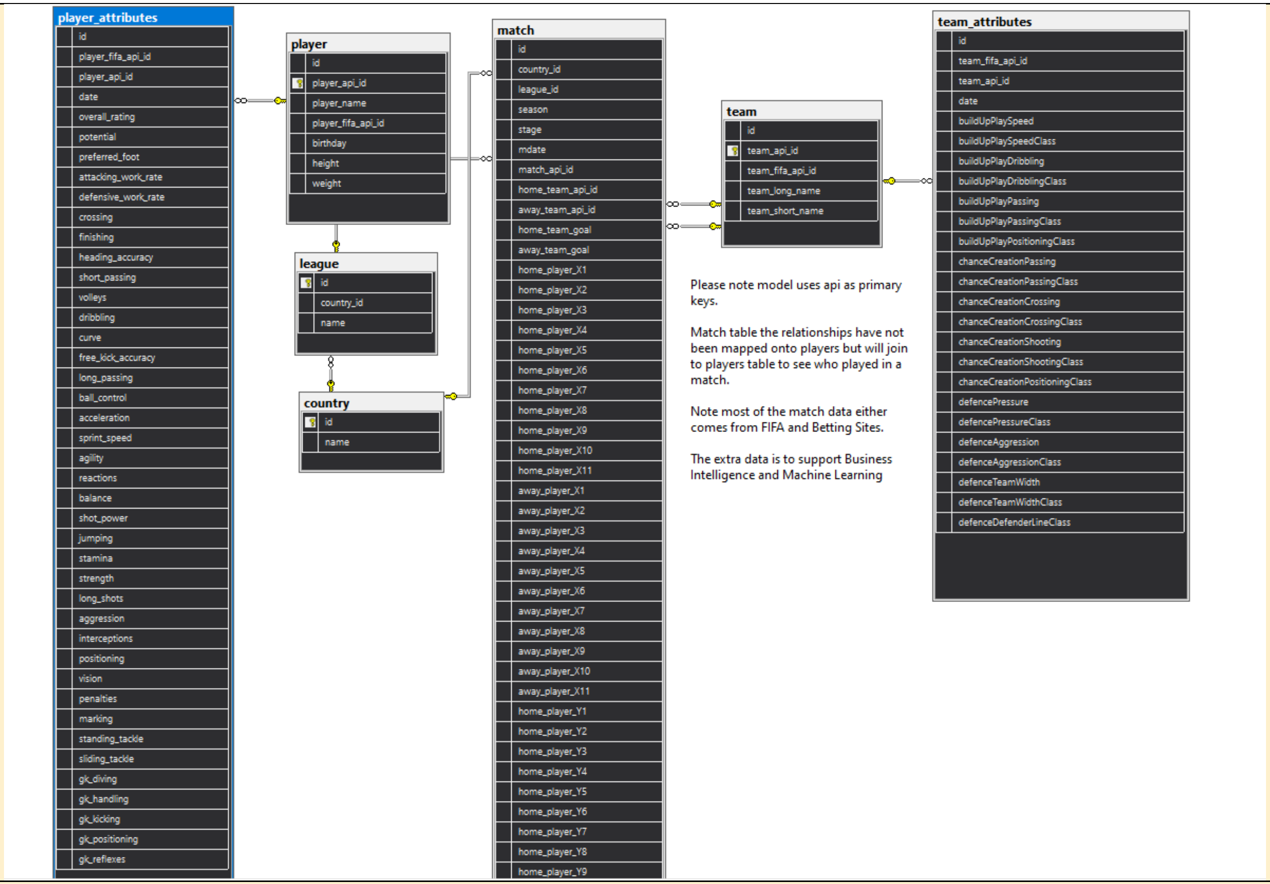
https://uk.indeed.com/viewjob?jk=a2b1bc089255c5e4&utm_campaign=google_jobs_apply&utm_source=google_jobs_apply&utm_medium=organic

T-SQL Server Database Overview

T-SQL SERVER DATABASE FOR DEMOS

I have investigated a European Top Leagues SQL Server database to write a range of tailored T-SQL queries aimed at gaining insights from the mass amount of football data provided. The European Top Leagues database contains many tables which contain football data relating to countries, leagues, matches, players and teams. The queries I have written are designed to meet user needs and support various use cases, from the performance analysis of players and teams to app integration. The objective of these queries is to demonstrate the transformation of raw match and player statistics into meaningful data that can be fed into web or mobile applications. Included in this document are examples of my best T-SQL demos to assist users working with the European Top Leagues database.

T-SQL SERVER DATABASE DIAGRAMS



Please note model uses api as primary keys.

Match table the relationships have not been mapped onto players but will join to players table to see who played in a match.

Note most of the match data either comes from FIFA and Betting Sites.

The extra data is to support Business Intelligence and Machine Learning

Main Tables of Interest for Supporting T-SQL Queries

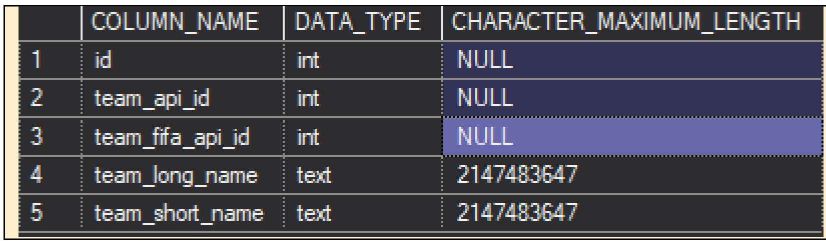
- The player table to be joined with the player_attributes table.
- The team table to be joined with the team_attributes table.
- The team table to be joined with the match table.

T-SQL SUPPORTING QUERIES

TSQL Demo Code Evidence/Results in SSMS

```
--SELECT * queries from the different tables in the EuroLeagues database.
--Used to select all data from every column and row from a specific table in the
EuroLeagues database.
SELECT * FROM country;
SELECT * FROM league;
SELECT * FROM match;
SELECT * FROM player;
SELECT * FROM player_attributes;
SELECT * FROM team;
SELECT * FROM team_attributes;
```

```
--Check the data types of all columns in the different tables stored in the
EuroLeagues database.
--Replace the TABLE_NAME string with the table that is needed for check.
SELECT COLUMN_NAME, DATA_TYPE, CHARACTER_MAXIMUM_LENGTH
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = 'team';
```



	COLUMN_NAME	DATA_TYPE	CHARACTER_MAXIMUM_LENGTH
1	id	int	NULL
2	team_api_id	int	NULL
3	team_fifa_api_id	int	NULL
4	team_long_name	text	2147483647
5	team_short_name	text	2147483647

T-SQL Part One: SQL Server Coding Basics (T-SQL03 to TSQL08)

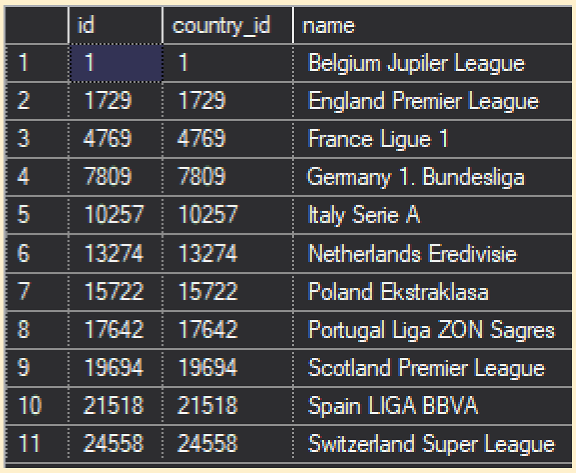
.sql File for TSQL03-08 Demos:	https://github.com/CarlBaines/Uni-Y1-SQL_Server_Portfolio_ICA

MODULE 3: WRITING SELECT QUERIES WITH SINGLE TABLE

DEMO 1: Writing Simple SELECT query

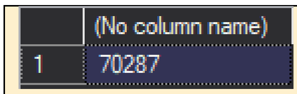
TSQL Demo Code Evidence/Results in SSMS

```
USE EuroLeagues
ALTER AUTHORIZATION ON DATABASE:: EuroLeagues TO sa
GO

--SELECT * queries from the different tables in the EuroLeagues database.
--Explanation: Used to select all data from every column and row from a specific
table in the EuroLeagues database.
SELECT * FROM league;
```



	id	country_id	name
1	1	1	Belgium Jupiler League
2	1729	1729	England Premier League
3	4769	4769	France Ligue 1
4	7809	7809	Germany 1. Bundesliga
5	10257	10257	Italy Serie A
6	13274	13274	Netherlands Eredivisie
7	15722	15722	Poland Ekstraklasa
8	17642	17642	Portugal Liga ZON Sagres
9	19694	19694	Scotland Premier League
10	21518	21518	Spain LIGA BBVA
11	24558	24558	Switzerland Super League

```
--User Story: Select the total number of goals scored from the EuroLeagues.match
table.
--Explanation: Simple SELECT query that creates a calculated column, calling the
sum function on the home_team_goal and away_team_goal columns.
SELECT SUM(home_team_goal + away_team_goal)
FROM match;
```



	(No column name)
1	70287

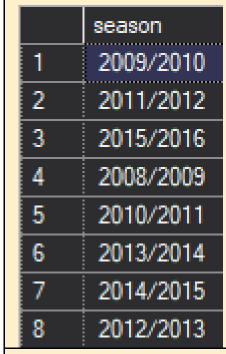
DEMO 2: Eliminating Duplicates with DISTINCT

TSQL Demo Code Evidence/Results in SSMS

```
--User Story: Eliminate duplicate seasons from the EuroLeagues.match table and
order them from earliest to latest.
--Explanation: This query uses a subquery and casts the season as a varchar type
(it was initially stored as a text value), so that it can work directly with
functions like LEFT().
--It extracts the starting years of the seasons (the first four characters), casts
them to an int and orders them.
SELECT season
FROM(
       SELECT DISTINCT CAST(season AS VARCHAR(MAX)) AS season
       FROM match
) AS season
ORDER BY CAST(LEFT(season, 4) AS INT);
```

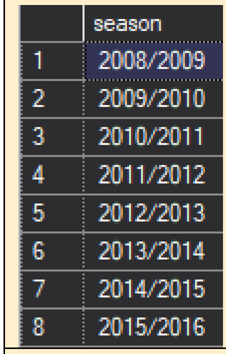
Result of the subquery:

```
SELECT DISTINCT CAST(season AS VARCHAR(MAX)) AS season
       FROM match
```



	season
1	2009/2010
2	2011/2012
3	2015/2016
4	2008/2009
5	2010/2011
6	2013/2014
7	2014/2015
8	2012/2013

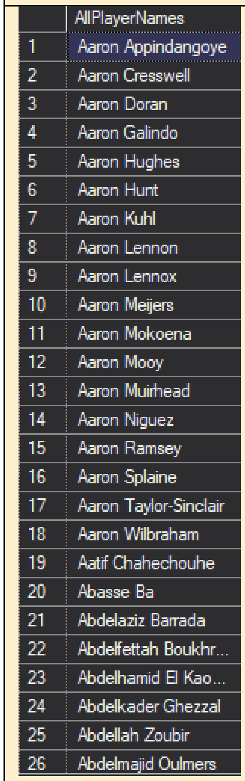
Result of the entire query:



	season
1	2008/2009
2	2009/2010
3	2010/2011
4	2011/2012
5	2012/2013
6	2013/2014
7	2014/2015
8	2015/2016

```
--User Story: Select unique player names from the EuroLeagues.player table and
stores them in a column called 'AllPlayerNames'.
```

```
SELECT DISTINCT CAST(player_name AS VARCHAR(MAX)) AS AllPlayerNames
FROM player;
```



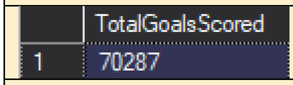
	AllPlayerNames
1	Aaron Appindangoye
2	Aaron Cresswell
3	Aaron Doran
4	Aaron Galindo
5	Aaron Hughes
6	Aaron Hunt
7	Aaron Kuhl
8	Aaron Lennon
9	Aaron Lennox
10	Aaron Meijers
11	Aaron Mokoena
12	Aaron Mooy
13	Aaron Muirhead
14	Aaron Niguez
15	Aaron Ramsey
16	Aaron Splaine
17	Aaron Taylor-Sinclair
18	Aaron Wilbraham
19	Aatif Chahechouhe
20	Abasse Ba
21	Abdelaziz Barrada
22	Abdelfettah Boukhr...
23	Abdelhamid El Kao...
24	Abdelkader Ghezzal
25	Abdellah Zoubir
26	Abdelmajid Oulmers

...

DEMO 3: Using Column and Table Aliases

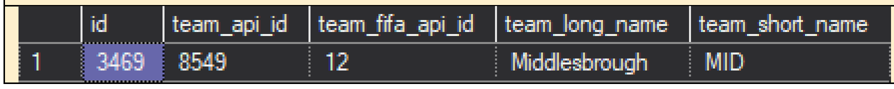
TSQL Demo Code Evidence/Results in SSMS

```
--Demo A3 Query Three
--Module 3: Using Column and Table Aliases Lesson
--User Story: Select the total number of goals scored from the EuroLeagues.match
table and assign the column the 'TotalGoalsScored' alias.
SELECT SUM(home_team_goal + away_team_goal) AS TotalGoalsScored
FROM match;
```



	TotalGoalsScored
1	70287

```
--Demo A4 Query Two
--Module 3: Using Column and Table Aliases Lesson
--User Story: Select all columns from the EuroLeagues.team table using the alias
'MiddlesbroughFCInfo', where the team_long_name is Middlesbrough.
SELECT id, team_api_id, team_fifa_api_id, team_long_name, team_short_name
FROM team AS MiddlesbroughFCInfo
WHERE CAST(team_long_name AS VARCHAR(MAX)) = 'Middlesbrough';
```

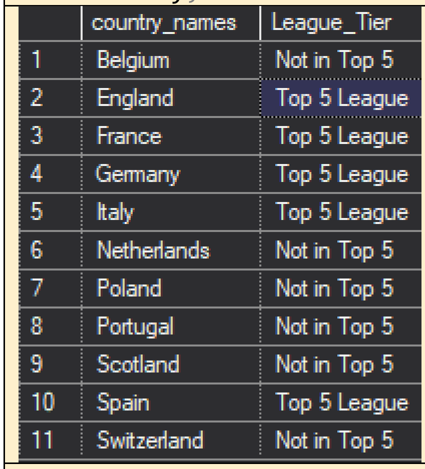


	id	team_api_id	team_fifa_api_id	team_long_name	team_short_name
1	3469	8549	12	Middlesbrough	MID

DEMO 4: Writing SIMPLE Case Expressions

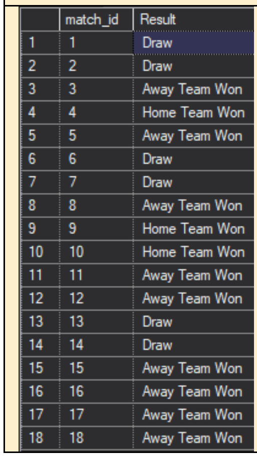
TSQL Demo Code Evidence/Results in SSMS

```sql
--Demo A4 Query One
--Module 4: Writing Simple CASE expressions
--Demo A4 Query One
--Module 4: Writing Simple CASE expressions
--User Story: Categorise countries by league tier.
--The name of the countries is casted as a varchar so that it can work directly
with functions.
SELECT CAST(name AS VARCHAR(MAX)) AS country_names,
       CASE
              WHEN CAST(name AS VARCHAR(MAX)) IN ('England', 'Spain', 'France',
'Germany', 'Italy') THEN 'Top 5 League'
              ELSE 'Not in Top 5'
       END AS League_Tier
FROM country;
```



	country_names	League_Tier
1	Belgium	Not in Top 5
2	England	Top 5 League
3	France	Top 5 League
4	Germany	Top 5 League
5	Italy	Top 5 League
6	Netherlands	Not in Top 5
7	Poland	Not in Top 5
8	Portugal	Not in Top 5
9	Scotland	Not in Top 5
10	Spain	Top 5 League
11	Switzerland	Not in Top 5

```sql
--Demo A4 Query Two
--Module 4: Writing Simple CASE expressions
--User Story: Determine the result of a match using the match table.
SELECT id AS match_id,
       CASE
              WHEN home_team_goal > away_team_goal THEN 'Home Team Won'
              WHEN home_team_goal < away_team_goal THEN 'Away Team Won'
              ELSE 'Draw'
       END AS Result
FROM match;
```



	match_id	Result
1	1	Draw
2	2	Draw
3	3	Away Team Won
4	4	Home Team Won
5	5	Away Team Won
6	6	Draw
7	7	Draw
8	8	Away Team Won
9	9	Home Team Won
10	10	Home Team Won
11	11	Away Team Won
12	12	Away Team Won
13	13	Draw
14	14	Draw
15	15	Away Team Won
16	16	Away Team Won
17	17	Away Team Won
18	18	Away Team Won

MODULE 4: JOINING AND QUERYING MULTIPLE TABLES

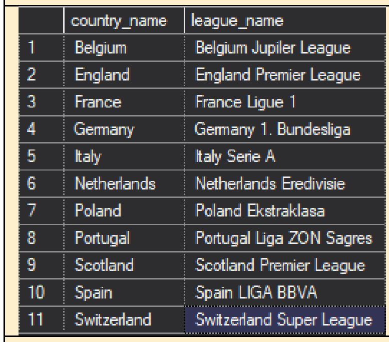
Why use Joining and Querying Multiple Tables?

Joining is especially useful as it allows the retrieval of data from two or more tables based on logical relationships between them. Querying data from multiple tables is equally useful as it allows for more complex data retrieval with more informative result sets.

DEMO 1: How to provide data from 2 related tables with a Join

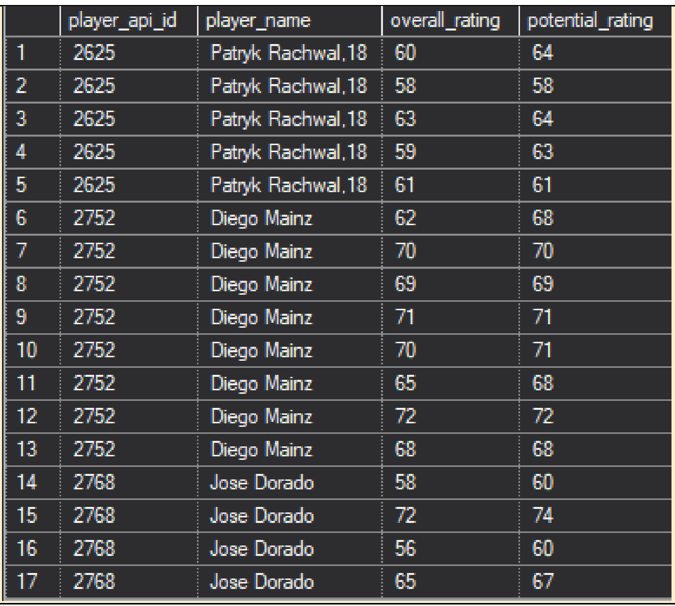
TSQL Demo Code Evidence/Results in SSMS

```
--Demo B1 Query One
--Module 4: How to provide data from 2 related tables with a Join.
--User Story: Select the league names associated with each country.
SELECT c.name AS country_name, l.name AS league_name
FROM country AS c
JOIN league AS l
ON c.id = l.country_id;
```



```
   country_name  league_name
1  Belgium       Belgium Jupiler League
2  England       England Premier League
3  France        France Ligue 1
4  Germany       Germany 1. Bundesliga
5  Italy         Italy Serie A
6  Netherlands   Netherlands Eredivisie
7  Poland        Poland Ekstraklasa
8  Portugal      Portugal Liga ZON Sagres
9  Scotland      Scotland Premier League
10 Spain         Spain LIGA BBVA
11 Switzerland   Switzerland Super League
```

```
--Demo B1 Query Two
--Module 4: How to provide data from 2 related tables with a Join.
SELECT DISTINCT p.player_api_id, CAST(p.player_name AS varchar(MAX)) AS
player_name, pa.overall_rating, pa.potential AS potential_rating
FROM player AS p
JOIN player_attributes AS pa
ON p.player_api_id = pa.player_api_id
ORDER BY p.player_api_id;
--Notice how there are many duplicate player names and ratings, this is because
each player has had multiple ratings assigned to them across many career dates.
```

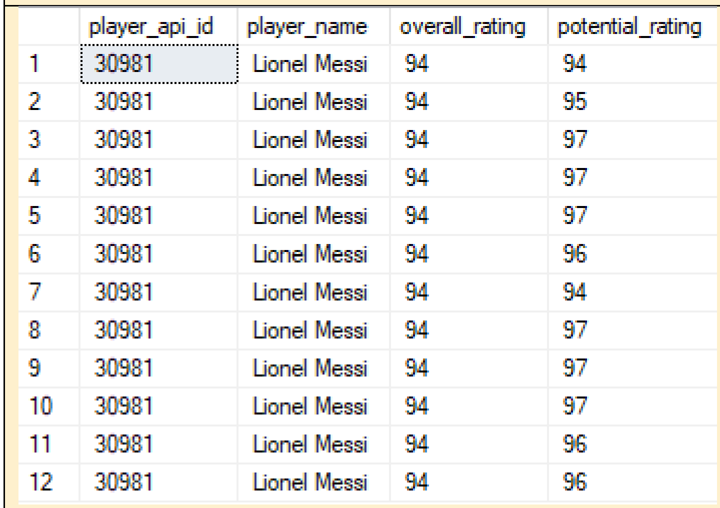


	player_api_id	player_name	overall_rating	potential_rating
1	2625	Patryk Rachwal,18	60	64
2	2625	Patryk Rachwal,18	58	58
3	2625	Patryk Rachwal,18	63	64
4	2625	Patryk Rachwal,18	59	63
5	2625	Patryk Rachwal,18	61	61
6	2752	Diego Mainz	62	68
7	2752	Diego Mainz	70	70
8	2752	Diego Mainz	69	69
9	2752	Diego Mainz	71	71
10	2752	Diego Mainz	70	71
11	2752	Diego Mainz	65	68
12	2752	Diego Mainz	72	72
13	2752	Diego Mainz	68	68
14	2768	Jose Dorado	58	60
15	2768	Jose Dorado	72	74
16	2768	Jose Dorado	56	60
17	2768	Jose Dorado	65	67

DEMO 2: How to query with inner joins

TSQL Demo Code Evidence/Results in SSMS

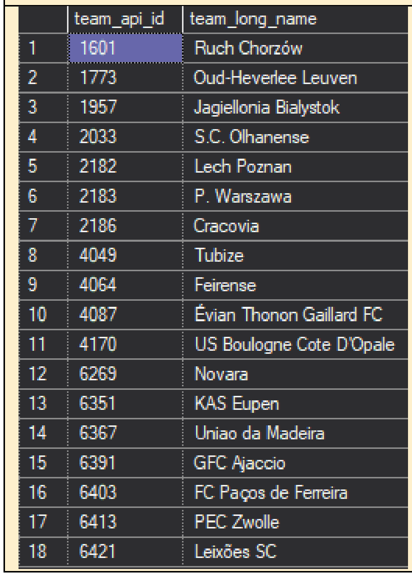
```
--Demo B2 Query One
--Module 4: How to query with inner joins.
--User Story: Select all the different ratings of the best player (the best player
has the highest overall and potential ratings)
--Lionel Messi.
SELECT p.player_api_id, CAST(p.player_name AS varchar(MAX)) AS player_name,
pa.overall_rating, pa.potential AS potential_rating
FROM player AS p
JOIN player_attributes AS pa
ON p.player_api_id = pa.player_api_id
WHERE pa.overall_rating = (SELECT MAX(pa.overall_rating) FROM player_attributes AS
pa)
ORDER BY p.player_api_id
```



	player_api_id	player_name	overall_rating	potential_rating
1	30981	Lionel Messi	94	94
2	30981	Lionel Messi	94	95
3	30981	Lionel Messi	94	97
4	30981	Lionel Messi	94	97
5	30981	Lionel Messi	94	97
6	30981	Lionel Messi	94	96
7	30981	Lionel Messi	94	94
8	30981	Lionel Messi	94	97
9	30981	Lionel Messi	94	97
10	30981	Lionel Messi	94	97
11	30981	Lionel Messi	94	96
12	30981	Lionel Messi	94	96

```
--Demo B2 Query Two
--Module 4: How to query with inner joins
--User Story: Join the match table with the team table to get the home and away
```

```
team names.
SELECT DISTINCT team_api_id, CAST(team_long_name AS varchar(MAX)) AS team_long_name
FROM team
JOIN match
ON home_team_api_id = team_api_id OR away_team_api_id = team_api_id
ORDER BY team_api_id;
```

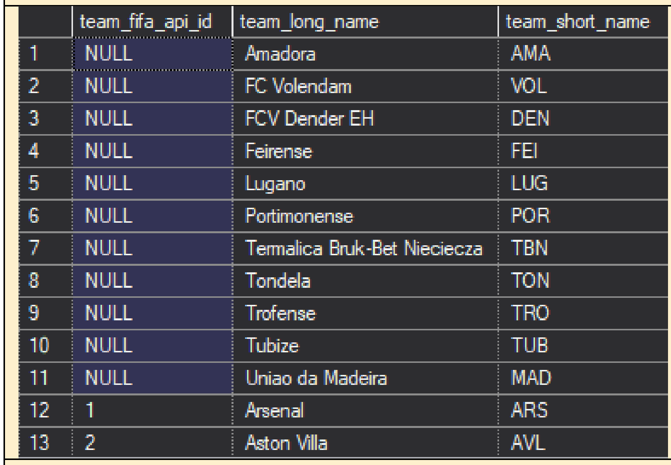


	team_api_id	team_long_name
1	1601	Ruch Chorzów
2	1773	Oud-Heverlee Leuven
3	1957	Jagiellonia Bialystok
4	2033	S.C. Olhanense
5	2182	Lech Poznan
6	2183	P. Warszawa
7	2186	Cracovia
8	4049	Tubize
9	4064	Feirense
10	4087	Évian Thonon Gaillard FC
11	4170	US Boulogne Cote D'Opale
12	6269	Novara
13	6351	KAS Eupen
14	6367	Uniao da Madeira
15	6391	GFC Ajaccio
16	6403	FC Paços de Ferreira
17	6413	PEC Zwolle
18	6421	Leixões SC

DEMO 3: How to query with outer joins

TSQL Demo Code Evidence/Results in SSMS

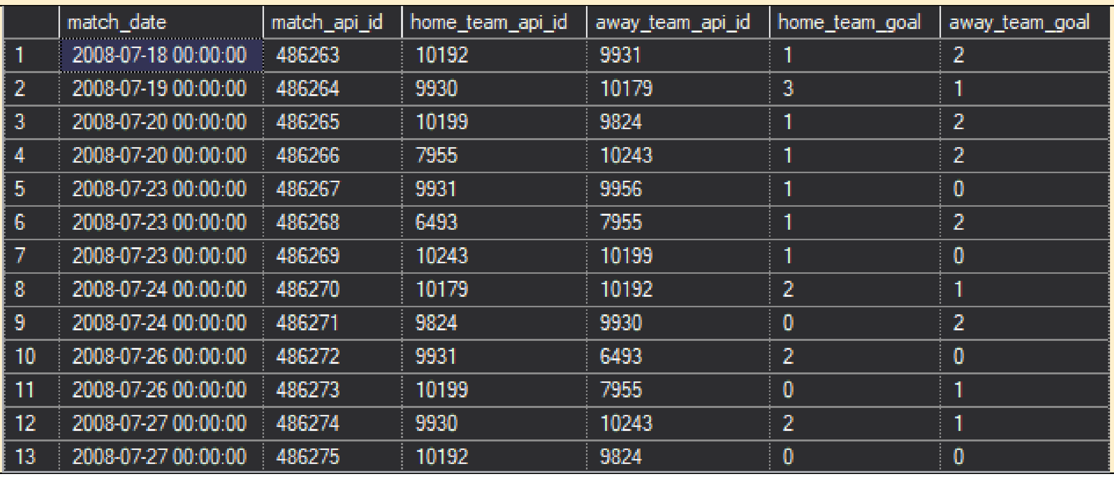
```
--Demo B3 Query One
--Module 4: How to query with outer joins
--User Story: full outer join between team and team_attributes to retrieve a
distinct list of all teams, including those with or without associated attribute
data.
SELECT DISTINCT t.team_fifa_api_id, CAST(t.team_long_name AS varchar(MAX)) AS
team_long_name, CAST(t.team_short_name AS varchar(MAX)) AS team_short_name
FROM team AS t
FULL OUTER JOIN team_attributes AS ta
ON t.team_fifa_api_id = ta.team_fifa_api_id;
```



	team_fifa_api_id	team_long_name	team_short_name
1	NULL	Amadora	AMA
2	NULL	FC Volendam	VOL
3	NULL	FCV Dender EH	DEN
4	NULL	Feirense	FEI
5	NULL	Lugano	LUG
6	NULL	Portimonense	POR
7	NULL	Termalica Bruk-Bet Nieciecza	TBN
8	NULL	Tondela	TON
9	NULL	Trofense	TRO
10	NULL	Tubize	TUB
11	NULL	Uniao da Madeira	MAD
12	1	Arsenal	ARS
13	2	Aston Villa	AVL

```
--Demo B3 Query Two
--Module 4: How to query with outer joins
--User Story: full outer join between match and team to retrieve a distinct list of
```

```
all matches, ensuring that match data is included even if team details are
duplicated or missing due to the join condition.
SELECT DISTINCT CAST(m.mdate AS varchar(MAX)) AS match_date, m.match_api_id,
m.home_team_api_id, m.away_team_api_id, m.home_team_goal, m.away_team_goal
FROM match AS m
FULL OUTER JOIN team AS t
ON m.home_team_api_id = t.team_api_id OR m.away_team_api_id = t.team_api_id;
```

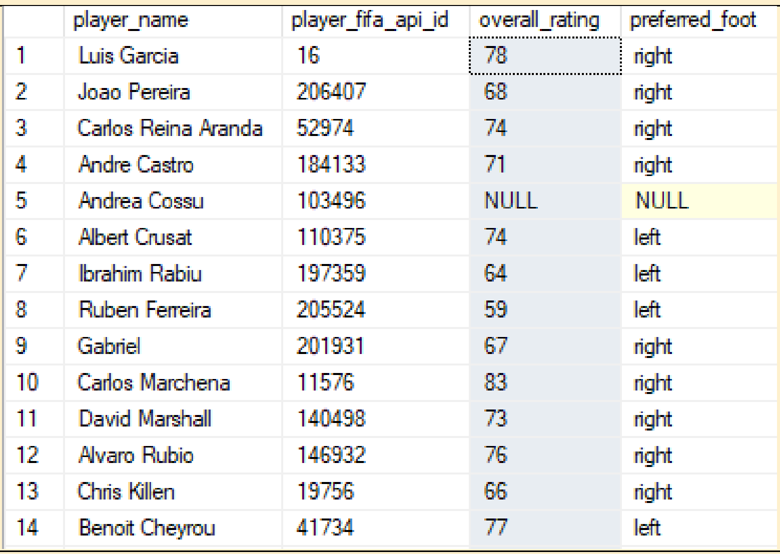


	match_date	match_api_id	home_team_api_id	away_team_api_id	home_team_goal	away_team_goal
1	2008-07-18 00:00:00	486263	10192	9931	1	2
2	2008-07-19 00:00:00	486264	9930	10179	3	1
3	2008-07-20 00:00:00	486265	10199	9824	1	2
4	2008-07-20 00:00:00	486266	7955	10243	1	2
5	2008-07-23 00:00:00	486267	9931	9956	1	0
6	2008-07-23 00:00:00	486268	6493	7955	1	2
7	2008-07-23 00:00:00	486269	10243	10199	1	0
8	2008-07-24 00:00:00	486270	10179	10192	2	1
9	2008-07-24 00:00:00	486271	9824	9930	0	2
10	2008-07-26 00:00:00	486272	9931	6493	2	0
11	2008-07-26 00:00:00	486273	10199	7955	0	1
12	2008-07-27 00:00:00	486274	9930	10243	2	1
13	2008-07-27 00:00:00	486275	10192	9824	0	0

DEMO 4: How to query with cross joins and self joins

TSQL Demo Code Evidence/Results in SSMS

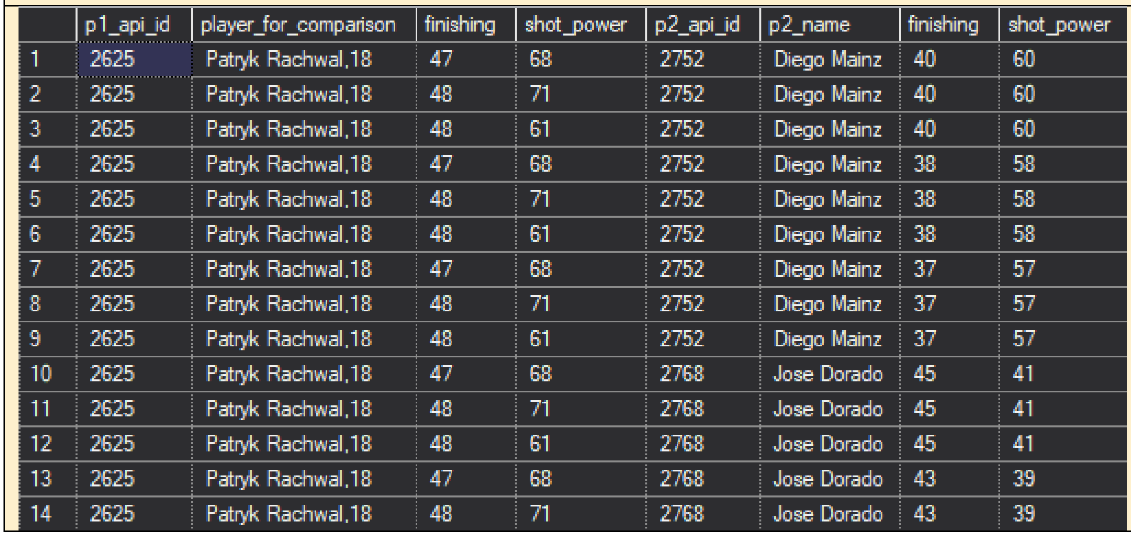
```
--Demo B4 Query One
--Module 4: How to query with cross and self joins
--Description: Retrieves distinct player names, their fifa API ids, overall
ratings, and preferred foot by cross joining the player and player_attributes
tables together.
SELECT DISTINCT CAST(p.player_name AS varchar(MAX)) AS player_name,
p.player_fifa_api_id, pa.overall_rating, CAST(pa.preferred_foot AS varchar(MAX)) AS
preferred_foot
FROM player AS p
CROSS JOIN player_attributes AS pa
WHERE p.player_api_id = pa.player_api_id;
```



	player_name	player_fifa_api_id	overall_rating	preferred_foot
1	Luis Garcia	16	78	right
2	Joao Pereira	206407	68	right
3	Carlos Reina Aranda	52974	74	right
4	Andre Castro	184133	71	right
5	Andrea Cossu	103496	NULL	NULL
6	Albert Crusat	110375	74	left
7	Ibrahim Rabiu	197359	64	left
8	Ruben Ferreira	205524	59	left
9	Gabriel	201931	67	right
10	Carlos Marchena	11576	83	right
11	David Marshall	140498	73	right
12	Alvaro Rubio	146932	76	right
13	Chris Killen	19756	66	right
14	Benoit Cheyrou	41734	77	left

```
SELECT DISTINCT TOP 100
```

```
          p1.player_api_id AS p1_api_id,
          CAST(p1_player.player_name AS varchar(MAX)) AS player_for_comparison,
          p1.finishing,
          p1.shot_power,
          p2.player_api_id AS p2_api_id,
          CAST(p2_player.player_name AS varchar(MAX)) AS p2_name,
          p2.finishing,
          p2.shot_power
FROM player_attributes p1
JOIN player_attributes p2
       ON p1.player_api_id = 2625 AND p1.player_api_id <> p2.player_api_id
JOIN player p1_player ON p1.player_api_id = p1_player.player_api_id
JOIN player p2_player ON p2.player_api_id = p2_player.player_api_id
ORDER BY p2.player_api_id;
```



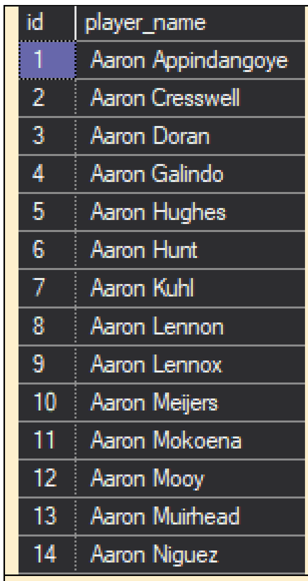
	p1_api_id	player_for_comparison	finishing	shot_power	p2_api_id	p2_name	finishing	shot_power
1	2625	Patryk Rachwal,18	47	68	2752	Diego Mainz	40	60
2	2625	Patryk Rachwal,18	48	71	2752	Diego Mainz	40	60
3	2625	Patryk Rachwal,18	48	61	2752	Diego Mainz	40	60
4	2625	Patryk Rachwal,18	47	68	2752	Diego Mainz	38	58
5	2625	Patryk Rachwal,18	48	71	2752	Diego Mainz	38	58
6	2625	Patryk Rachwal,18	48	61	2752	Diego Mainz	38	58
7	2625	Patryk Rachwal,18	47	68	2752	Diego Mainz	37	57
8	2625	Patryk Rachwal,18	48	71	2752	Diego Mainz	37	57
9	2625	Patryk Rachwal,18	48	61	2752	Diego Mainz	37	57
10	2625	Patryk Rachwal,18	47	68	2768	Jose Dorado	45	41
11	2625	Patryk Rachwal,18	48	71	2768	Jose Dorado	45	41
12	2625	Patryk Rachwal,18	48	61	2768	Jose Dorado	45	41
13	2625	Patryk Rachwal,18	47	68	2768	Jose Dorado	43	39
14	2625	Patryk Rachwal,18	48	71	2768	Jose Dorado	43	39

MODULE 5: SORTING AND FILTERING DATA

DEMO 1: How to Sort Data

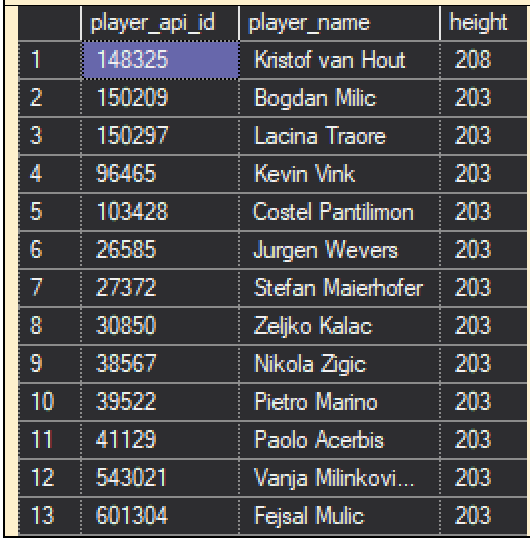
TSQL Demo Code Evidence/Results in SSMS

```
--Demo B5 Query One
--Module 5: How to Sort Data
--Description: Select id and player names from the EuroLeagues.player table,
ordering the id in ascending order.
--Simple SELECT query with ORDER by clause.
SELECT id, player_name
FROM player
ORDER BY id ASC;
```



id	player_name
1	Aaron Appindangoye
2	Aaron Cresswell
3	Aaron Doran
4	Aaron Galindo
5	Aaron Hughes
6	Aaron Hunt
7	Aaron Kuhl
8	Aaron Lennon
9	Aaron Lennox
10	Aaron Meijers
11	Aaron Mokoena
12	Aaron Mooy
13	Aaron Muirhead
14	Aaron Niguez

```
--Demo B5 Query Two
--Module 5: How to Sort Data
--Description: Selects the player_api_id, name and height from the
EuroLeagues.player table, ordering the players by height in descending order.
SELECT player_api_id, player_name, height
FROM player
ORDER BY height DESC;
```



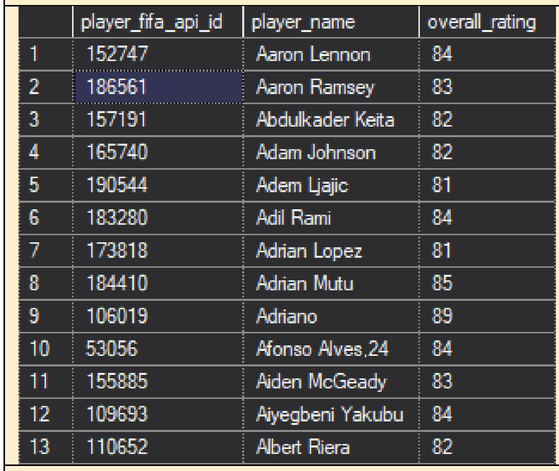
	player_api_id	player_name	height
1	148325	Kristof van Hout	208
2	150209	Bogdan Milic	203
3	150297	Lacina Traore	203
4	96465	Kevin Vink	203
5	103428	Costel Pantilimon	203
6	26585	Jurgen Wevers	203
7	27372	Stefan Maierhofer	203
8	30850	Zeljko Kalac	203
9	38567	Nikola Zigic	203
10	39522	Pietro Marino	203
11	41129	Paolo Acerbis	203
12	543021	Vanja Milinkovi...	203
13	601304	Fejsal Mulic	203

DEMO 2: How to Filter Data with Predicates

TSQL Demo Code Evidence/Results in SSMS

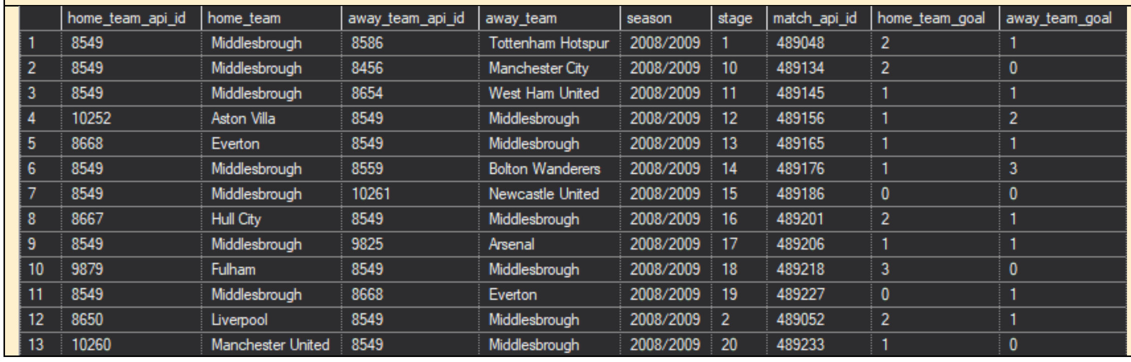
```
--Demo C2 Query One
--Module 5: How to filter data with predicates.
--Description: Retrieves a list of distinct players that have an overall rating
greater than 80. Each player only appears once with their highest rating.
SELECT DISTINCT pa.player_fifa_api_id, CAST(p.player_name AS varchar(MAX)) AS
player_name, MAX(pa.overall_rating) AS overall_rating
FROM player_attributes AS pa
JOIN player AS p
ON p.player_fifa_api_id = pa.player_fifa_api_id
WHERE overall_rating > 80
```

```sql
GROUP BY pa.player_fifa_api_id, CAST(p.player_name AS varchar(MAX));
```



	player_fifa_api_id	player_name	overall_rating
1	152747	Aaron Lennon	84
2	186561	Aaron Ramsey	83
3	157191	Abdulkader Keita	82
4	165740	Adam Johnson	82
5	190544	Adem Ljajic	81
6	183280	Adil Rami	84
7	173818	Adrian Lopez	81
8	184410	Adrian Mutu	85
9	106019	Adriano	89
10	53056	Afonso Alves,24	84
11	155885	Aiden McGeady	83
12	109693	Aiyegbeni Yakubu	84
13	110652	Albert Riera	82

```sql
--Demo C2 Query Two
--Module 5: How to filter data with predicates.
--Description: The query retrieves all match details involving Middlesbrough,
including the teams they played, the season (as well as its stage), the matchID and
the goals scored.
SELECT
       home_team.team_api_id AS home_team_api_id,
       CAST(home_team.team_long_name AS VARCHAR(MAX)) AS home_team,
       away_team.team_api_id AS away_team_api_id,
       CAST(away_team.team_long_name AS VARCHAR(MAX)) AS away_team,
       CAST(m.season AS VARCHAR(MAX)) AS season,
       m.stage,
       m.match_api_id,
       m.home_team_goal,
       m.away_team_goal
FROM match AS m
JOIN team AS home_team ON home_team.team_api_id = m.home_team_api_id
JOIN team AS away_team ON away_team.team_api_id = m.away_team_api_id
WHERE
       CAST(home_team.team_long_name AS varchar(MAX)) = 'Middlesbrough'
       OR CAST(away_team.team_long_name AS varchar(MAX)) = 'Middlesbrough';
```



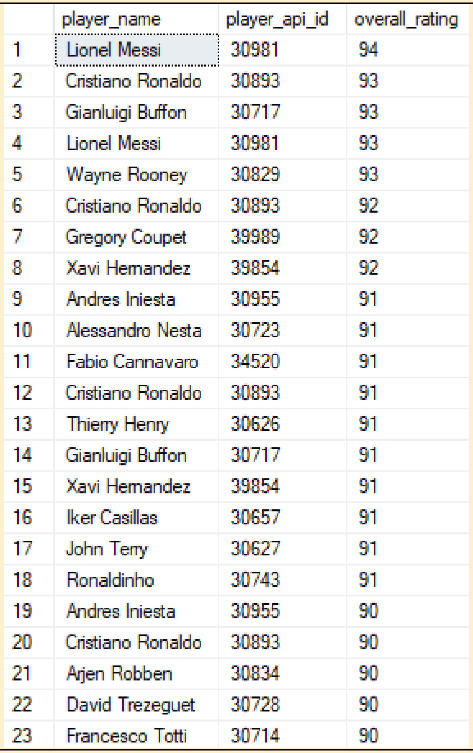
	home_team_api_id	home_team	away_team_api_id	away_team	season	stage	match_api_id	home_team_goal	away_team_goal
1	8549	Middlesbrough	8586	Tottenham Hotspur	2008/2009	1	489048	2	1
2	8549	Middlesbrough	8456	Manchester City	2008/2009	10	489134	2	0
3	8549	Middlesbrough	8654	West Ham United	2008/2009	11	489145	1	1
4	10252	Aston Villa	8549	Middlesbrough	2008/2009	12	489156	1	2
5	8668	Everton	8549	Middlesbrough	2008/2009	13	489165	1	1
6	8549	Middlesbrough	8559	Bolton Wanderers	2008/2009	14	489176	1	3
7	8549	Middlesbrough	10261	Newcastle United	2008/2009	15	489186	0	0
8	8667	Hull City	8549	Middlesbrough	2008/2009	16	489201	2	1
9	8549	Middlesbrough	9825	Arsenal	2008/2009	17	489206	1	1
10	9879	Fulham	8549	Middlesbrough	2008/2009	18	489218	3	0
11	8549	Middlesbrough	8668	Everton	2008/2009	19	489227	0	1
12	8650	Liverpool	8549	Middlesbrough	2008/2009	2	489052	2	1
13	10260	Manchester United	8549	Middlesbrough	2008/2009	20	489233	1	0

DEMO 3: How to Filter Data with TOP and OFFSET-FETCH

TSQL Demo Code Evidence/Results in SSMS

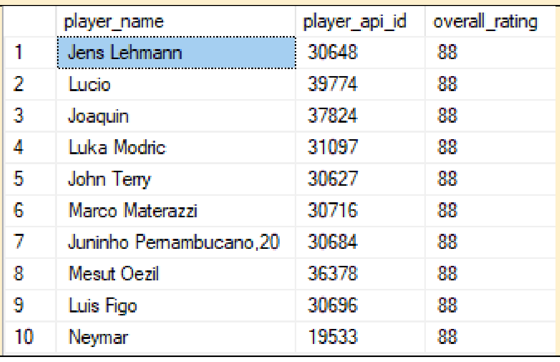
```sql
--Demo C3 Query One
--Module 5: How to filter data with TOP and OFFSET-FETCH.
--Description: Selects top 100 players and their api ids from the players table,
joining with the player attributes table to get their overall ratings.
```

```sql
--Their overall ratings are ordered in descending order.
SELECT DISTINCT TOP 100 CAST(p.player_name AS nvarchar(MAX)) AS player_name,
p.player_api_id, ISNULL(pa.overall_rating, '0') AS overall_rating
FROM player AS p
JOIN player_attributes AS pa
ON p.player_api_id = pa.player_api_id
GROUP BY CAST(p.player_name AS nvarchar(MAX)), p.player_api_id, pa.overall_rating
ORDER BY overall_rating DESC
```



	player_name	player_api_id	overall_rating
1	Lionel Messi	30981	94
2	Cristiano Ronaldo	30893	93
3	Gianluigi Buffon	30717	93
4	Lionel Messi	30981	93
5	Wayne Rooney	30829	93
6	Cristiano Ronaldo	30893	92
7	Gregory Coupet	39989	92
8	Xavi Hernandez	39854	92
9	Andres Iniesta	30955	91
10	Alessandro Nesta	30723	91
11	Fabio Cannavaro	34520	91
12	Cristiano Ronaldo	30893	91
13	Thierry Henry	30626	91
14	Gianluigi Buffon	30717	91
15	Xavi Hernandez	39854	91
16	Iker Casillas	30657	91
17	John Terry	30627	91
18	Ronaldinho	30743	91
19	Andres Iniesta	30955	90
20	Cristiano Ronaldo	30893	90
21	Arjen Robben	30834	90
22	David Trezeguet	30728	90
23	Francesco Totti	30714	90

```sql
--Demo C3 Query Two
--Module 5: How to filter data with TOP and OFFSET-FETCH
--Description: Selects the bottom 10 of the top 100 players with their api ids from
the players table, joining with the players attributes table to get their overall
ratings.
SELECT DISTINCT CAST(p.player_name AS nvarchar(MAX)) AS player_name,
p.player_api_id, ISNULL(pa.overall_rating, '0') AS overall_rating
FROM player AS p
JOIN player_attributes AS pa
ON p.player_api_id = pa.player_api_id
GROUP BY CAST(p.player_name AS nvarchar(MAX)), p.player_api_id, pa.overall_rating
ORDER BY overall_rating DESC
OFFSET 90 ROWS
FETCH NEXT 10 ROWS ONLY;
```

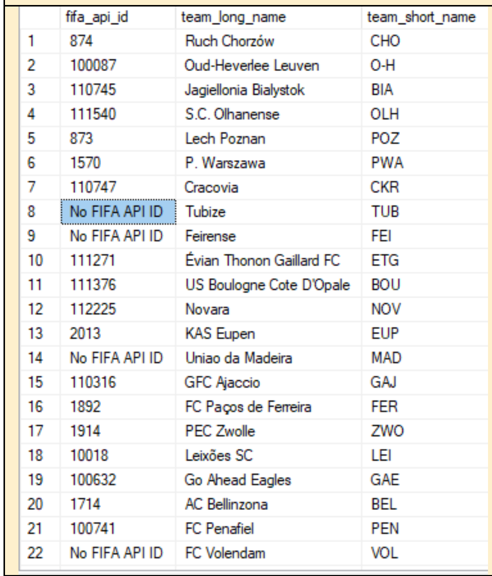


	player_name	player_api_id	overall_rating
1	Jens Lehmann	30648	88
2	Lucio	39774	88
3	Joaquin	37824	88
4	Luka Modric	31097	88
5	John Terry	30627	88
6	Marco Materazzi	30716	88
7	Juninho Pernambucano,20	30684	88
8	Mesut Oezil	36378	88
9	Luis Figo	30696	88
10	Neymar	19533	88

DEMO 4: How to work with Unknown Values

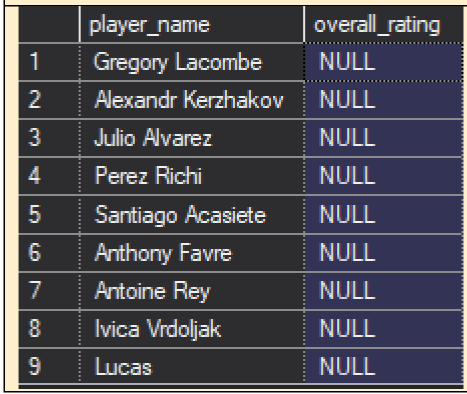
TSQL Demo Code Evidence/Results in SSMS

```
--Demo C4 Query One
--Module 5: How to work with unknown values
--Description: Selects the team_fifa_api_id, the long name and short name of teams
from the teams table.
--If the team_fifa_api_id is null, the string 'No FIFA API ID' is replaced in place
of the null value.
SELECT ISNULL(CAST(TRY_CAST(team_fifa_api_id AS INT) AS VARCHAR(255)), 'No FIFA API
ID') AS fifa_api_id, team_long_name, team_short_name
FROM team;
```



	fifa_api_id	team_long_name	team_short_name
1	874	Ruch Chorzów	CHO
2	100087	Oud-Heverlee Leuven	O-H
3	110745	Jagiellonia Bialystok	BIA
4	111540	S.C. Olhanense	OLH
5	873	Lech Poznan	POZ
6	1570	P. Warszawa	PWA
7	110747	Cracovia	CKR
8	No FIFA API ID	Tubize	TUB
9	No FIFA API ID	Feirense	FEI
10	111271	Évian Thonon Gaillard FC	ETG
11	111376	US Boulogne Cote D'Opale	BOU
12	112225	Novara	NOV
13	2013	KAS Eupen	EUP
14	No FIFA API ID	Uniao da Madeira	MAD
15	110316	GFC Ajaccio	GAJ
16	1892	FC Paços de Ferreira	FER
17	1914	PEC Zwolle	ZWO
18	10018	Leixões SC	LEI
19	100632	Go Ahead Eagles	GAE
20	1714	AC Bellinzona	BEL
21	100741	FC Penafiel	PEN
22	No FIFA API ID	FC Volendam	VOL

```
--Demo C4 Query Two
--Module 5: How to work with unknown values
--Description: Returns a list of player names with missing overall ratings.
SELECT p.player_name, pa.overall_rating
FROM player AS p
JOIN player_attributes AS pa
ON p.player_api_id = pa.player_api_id
WHERE pa.overall_rating IS NULL;
```



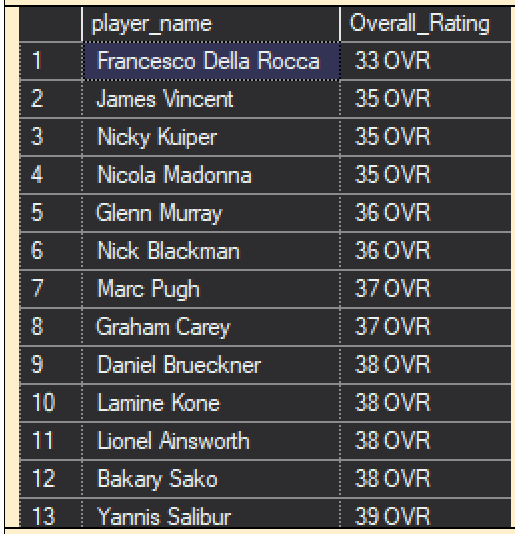
	player_name	overall_rating
1	Gregory Lacombe	NULL
2	Alexandr Kerzhakov	NULL
3	Julio Alvarez	NULL
4	Perez Richi	NULL
5	Santiago Acasiete	NULL
6	Anthony Favre	NULL
7	Antoine Rey	NULL
8	Ivica Vrdoljak	NULL
9	Lucas	NULL

MODULE 6: WORKING WITH DATA TYPES

DEMO 1: Working with Data Type examples

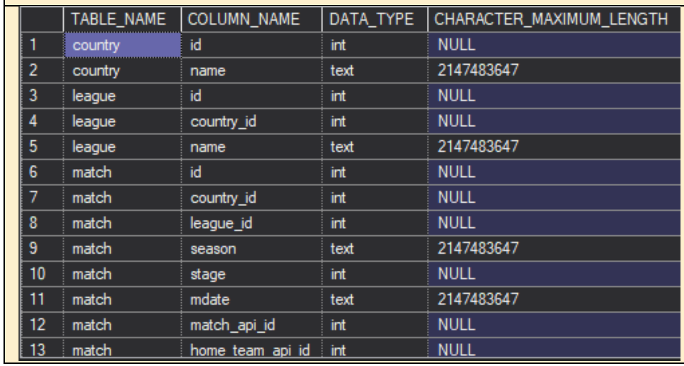
TSQL Demo Code Evidence/Results in SSMS

```
--Demo D1 Query One
--Module 6: Working with data types examples
--Description: This query demonstrates working with data types by casting numeric
and text fields using CAST and handling null values with ISNULL.
--The query selects distinct player names and their overall ratings, ordered in
ascending order. If a rating is null, it displays 'N/A'.
SELECT DISTINCT
          CAST(p.player_name AS VARCHAR) AS player_name,
          ISNULL(CAST(pa.overall_rating AS VARCHAR), 'N/A') + ' OVR' AS
Overall_Rating
FROM player_attributes pa
JOIN player p ON pa.player_api_id = p.player_api_id
ORDER BY Overall_Rating ASC;
```



	player_name	Overall_Rating
1	Francesco Della Rocca	33 OVR
2	James Vincent	35 OVR
3	Nicky Kuiper	35 OVR
4	Nicola Madonna	35 OVR
5	Glenn Murray	36 OVR
6	Nick Blackman	36 OVR
7	Marc Pugh	37 OVR
8	Graham Carey	37 OVR
9	Daniel Brueckner	38 OVR
10	Lamine Kone	38 OVR
11	Lionel Ainsworth	38 OVR
12	Bakary Sako	38 OVR
13	Yannis Salibur	39 OVR

```
--Demo D1 Query Two
--Module 6: Working with data types examples
--Description: This query demonstrates working with data types as it selects all
columns in the database, ordered by table_name
--and displays the data type and max character length of each.
SELECT TABLE_NAME, COLUMN_NAME, DATA_TYPE, CHARACTER_MAXIMUM_LENGTH
FROM INFORMATION_SCHEMA.COLUMNS
ORDER BY TABLE_NAME;
```



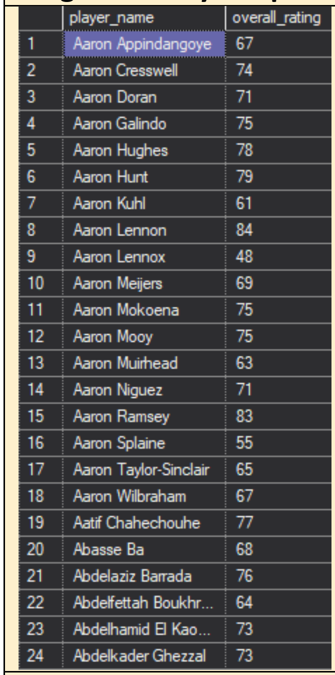
	TABLE_NAME	COLUMN_NAME	DATA_TYPE	CHARACTER_MAXIMUM_LENGTH
1	country	id	int	NULL
2	country	name	text	2147483647
3	league	id	int	NULL
4	league	country_id	int	NULL
5	league	name	text	2147483647
6	match	id	int	NULL
7	match	country_id	int	NULL
8	match	league_id	int	NULL
9	match	season	text	2147483647
10	match	stage	int	NULL
11	match	mdate	text	2147483647
12	match	match_api_id	int	NULL
13	match	home_team_api_id	int	NULL

DEMO 2: Working with Character Data

TSQL Demo Code Evidence/Results in SSMS

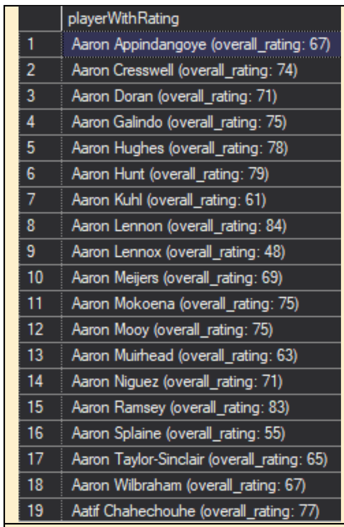
```sql
--Demo D2 Query One
--Module 6: Working with Character Data
--Original query I wanted to concatenate.
SELECT DISTINCT
       CAST(p.player_name AS varchar(MAX)) AS player_name,
       MAX(pa.overall_rating) AS overall_rating
FROM player AS p
JOIN player_attributes AS pa
       ON p.player_api_id = pa.player_api_id
GROUP BY CAST(p.player_name AS varchar(MAX));

--Description: This query returns a list of unique players from the player table
alongside their highest overall rating
--from the player_attributes table, formatted as a single string.
SELECT DISTINCT
       CONCAT(
              CAST(p.player_name AS varchar(MAX)),
              N' (overall_rating: ',
              CAST(MAX(pa.overall_rating) AS NVARCHAR),
              N')'
       ) AS playerWithRating
FROM player AS p
JOIN player_attributes AS pa
       ON p.player_api_id = pa.player_api_id
GROUP BY CAST(p.player_name AS varchar(MAX));
```

Original Query Output



	player_name	overall_rating
1	Aaron Appindangoye	67
2	Aaron Cresswell	74
3	Aaron Doran	71
4	Aaron Galindo	75
5	Aaron Hughes	78
6	Aaron Hunt	79
7	Aaron Kuhl	61
8	Aaron Lennon	84
9	Aaron Lennox	48
10	Aaron Meijers	69
11	Aaron Mokoena	75
12	Aaron Mooy	75
13	Aaron Muirhead	63
14	Aaron Niguez	71
15	Aaron Ramsey	83
16	Aaron Splaine	55
17	Aaron Taylor-Sinclair	65
18	Aaron Wilbraham	67
19	Aatif Chahechouhe	77
20	Abasse Ba	68
21	Abdelaziz Barrada	76
22	Abdelfettah Boukhr...	64
23	Abdelhamid El Kao...	73
24	Abdelkader Ghezzal	73

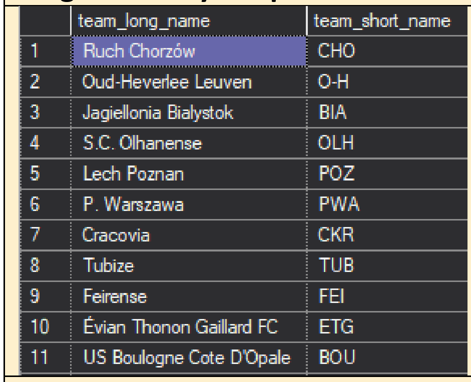
Query Output with Concatenation



	playerWithRating
1	Aaron Appindangoye (overall_rating: 67)
2	Aaron Cresswell (overall_rating: 74)
3	Aaron Doran (overall_rating: 71)
4	Aaron Galindo (overall_rating: 75)
5	Aaron Hughes (overall_rating: 78)
6	Aaron Hunt (overall_rating: 79)
7	Aaron Kuhl (overall_rating: 61)
8	Aaron Lennon (overall_rating: 84)
9	Aaron Lennox (overall_rating: 48)
10	Aaron Meijers (overall_rating: 69)
11	Aaron Mokoena (overall_rating: 75)
12	Aaron Mooy (overall_rating: 75)
13	Aaron Muirhead (overall_rating: 63)
14	Aaron Niguez (overall_rating: 71)
15	Aaron Ramsey (overall_rating: 83)
16	Aaron Splaine (overall_rating: 55)
17	Aaron Taylor-Sinclair (overall_rating: 65)
18	Aaron Wilbraham (overall_rating: 67)
19	Aatif Chahechouhe (overall_rating: 77)

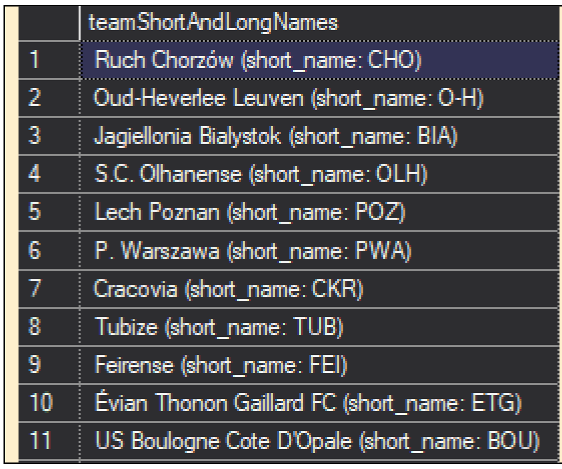
```sql
--Demo D2 Query Two
--Module 6: Working with Character Data
--Original Query I wanted to concatenate.
SELECT team_long_name, team_short_name
FROM team;

--Description: This query returns a list of team short and long names, formatted as
a single string.
SELECT
       CONCAT(
              team_long_name,
              N' (short_name: ',
              team_short_name,
              N')'
       ) AS teamShortAndLongNames
FROM team;
```

Original Query Output



	team_long_name	team_short_name
1	Ruch Chorzów	CHO
2	Oud-Heverlee Leuven	O-H
3	Jagiellonia Bialystok	BIA
4	S.C. Olhanense	OLH
5	Lech Poznan	POZ
6	P. Warszawa	PWA
7	Cracovia	CKR
8	Tubize	TUB
9	Feirense	FEI
10	Évian Thonon Gaillard FC	ETG
11	US Boulogne Cote D'Opale	BOU

Query Output with Concatenation

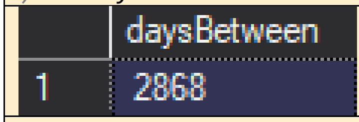


	teamShortAndLongNames
1	Ruch Chorzów (short_name: CHO)
2	Oud-Heverlee Leuven (short_name: O-H)
3	Jagiellonia Bialystok (short_name: BIA)
4	S.C. Olhanense (short_name: OLH)
5	Lech Poznan (short_name: POZ)
6	P. Warszawa (short_name: PWA)
7	Cracovia (short_name: CKR)
8	Tubize (short_name: TUB)
9	Feirense (short_name: FEI)
10	Évian Thonon Gaillard FC (short_name: ETG)
11	US Boulogne Cote D'Opale (short_name: BOU)

DEMO 3: Working with Date and Time Data

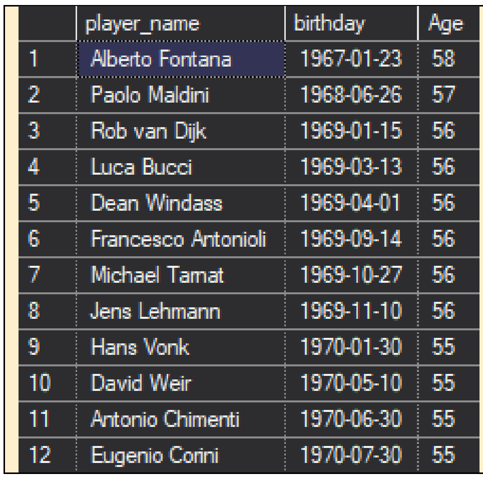
TSQL Demo Code Evidence/Results in SSMS

```
--Demo D3 Query One
--Module 6: Working with Date and Time Data
--Description: This query returns the difference between the first match date and
the last match date stored in the match table.
SELECT DATEDIFF(
       DAY,
       (SELECT TOP 1 CAST(mdate AS varchar(MAX)) AS mdate FROM match ORDER BY mdate
ASC),
       (SELECT TOP 1 CAST(mdate AS varchar(MAX)) AS mdate FROM match ORDER BY mdate
DESC)
) AS daysBetween
```



	daysBetween
1	2868

```
--Demo D3 Query Two
--Module 6: Working with Date and Time Data
--Description: This query returns all player names and their birthdays from the
player table along with their age. The query results are ordered by the oldest
birthday.
--The age is calculated from the birthday datetime values by using the DATEDIFF
function.
--The birthday column is converted from a text type and is first casted to a
varchar, so that it can then be casted to a date type, since SQL server
--does not allow for text types to be converted straight to a date/datetime type.
SELECT player_name,
       CAST(CAST(birthday AS varchar(MAX)) AS DATE) AS birthday,
       DATEDIFF(YEAR, CAST(CAST(birthday AS varchar(MAX)) AS DATE), '2025-04-16')
as Age
FROM player
ORDER BY birthday;
```



	player_name	birthday	Age
1	Alberto Fontana	1967-01-23	58
2	Paolo Maldini	1968-06-26	57
3	Rob van Dijk	1969-01-15	56
4	Luca Bucci	1969-03-13	56
5	Dean Windass	1969-04-01	56
6	Francesco Antonioli	1969-09-14	56
7	Michael Tarnat	1969-10-27	56
8	Jens Lehmann	1969-11-10	56
9	Hans Vonk	1970-01-30	55
10	David Weir	1970-05-10	55
11	Antonio Chimenti	1970-06-30	55
12	Eugenio Corini	1970-07-30	55

MODULE 7: USING DML TO MODIFY DATA

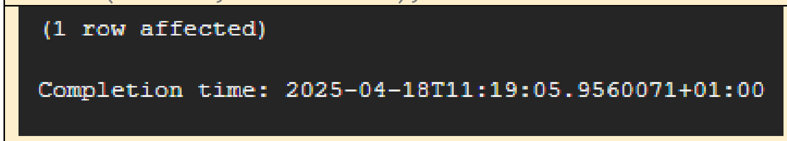
Why use DML to modify data?

DML is the short name for Data Manipulation Language which deals with data manipulation. Examples of DML in SQL include statements such as SELECT, INSERT, UPDATE and DELETE etc, which are used to store, modify, retrieve, delete and update data within a database.

DEMO 1: Adding Data to Tables

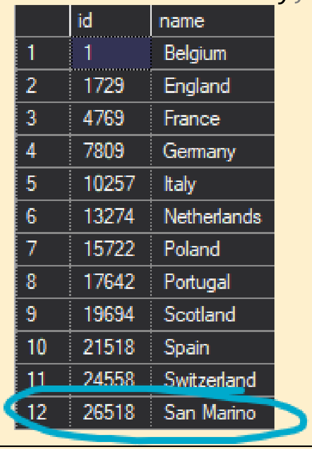
TSQL Demo Code Evidence/Results in SSMS

```sql
--Demo D4 Query One
--Module 7: Using DML to Modify Data.
--Description: Add the country San Marino to the countries table with a designated ID.
INSERT INTO country (id, name)
VALUES('26518','San Marino');
```



```
(1 row affected)

Completion time: 2025-04-18T11:19:05.9560071+01:00
```

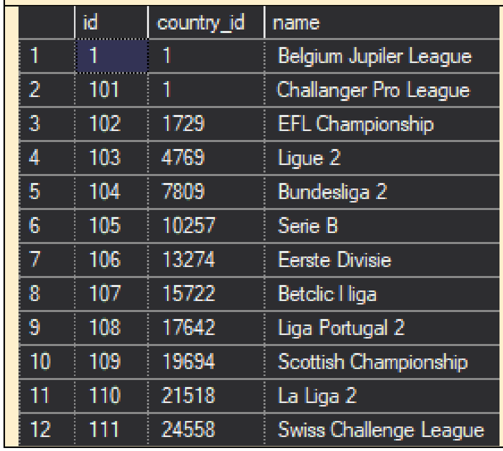
```sql
--Using the * wildcard to return all rows and values from the country table.
SELECT * FROM country;
```



	id	name
1	1	Belgium
2	1729	England
3	4769	France
4	7809	Germany
5	10257	Italy
6	13274	Netherlands
7	15722	Poland
8	17642	Portugal
9	19694	Scotland
10	21518	Spain
11	24558	Switzerland
12	26518	San Marino

```sql
--Demo D4 Query Two
```

```sql
--Module 7: Adding data to tables using DML.
--Description: Adding the second tier leagues of each country into the league
table.
INSERT INTO league(id, country_id, name)
VALUES
('101','1','Challanger Pro League'),
('102','1729','EFL Championship'),
('103','4769','Ligue 2'),
('104','7809','Bundesliga 2'),
('105','10257','Serie B'),
('106','13274','Eerste Divisie'),
('107','15722','Betclic l liga'),
('108','17642','Liga Portugal 2'),
('109','19694','Scottish Championship'),
('110','21518','La Liga 2'),
('111','24558','Swiss Challenge League');
```



	id	country_id	name
1	1	1	Belgium Jupiler League
2	101	1	Challanger Pro League
3	102	1729	EFL Championship
4	103	4769	Ligue 2
5	104	7809	Bundesliga 2
6	105	10257	Serie B
7	106	13274	Eerste Divisie
8	107	15722	Betclic I liga
9	108	17642	Liga Portugal 2
10	109	19694	Scottish Championship
11	110	21518	La Liga 2
12	111	24558	Swiss Challenge League

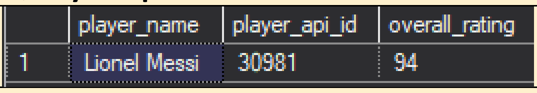
DEMO 2: Modifying and Removing Data

TSQL Demo Code Evidence/Results in SSMS

```sql
--Demo E1 Query One
--Module 7: Modifying and Removing Data
--The query which returns a result I want to update.
--It selects the best player based on highest overall_rating, returning the name,
api id, overall_rating, with their dribbling and ball control statistics.
SELECT DISTINCT TOP 1 CAST(p.player_name AS nvarchar(MAX)) AS player_name,
p.player_api_id, pa.overall_rating AS overall_rating, pa.dribbling, pa.ball_control
FROM player AS p
JOIN player_attributes AS pa
ON p.player_api_id = pa.player_api_id
GROUP BY CAST(p.player_name AS nvarchar(MAX)), p.player_api_id, pa.overall_rating,
pa.dribbling, pa.ball_control
ORDER BY overall_rating DESC

--Description: This is the query I used to update the ball control statistic of the
best player (Lionel Messi), based on highest overall_rating.
--It finds Messi and increases his ball_control stat by 3.
WITH TopPlayer AS (
    SELECT TOP 1 pa.player_api_id AS player_attribute_id
    FROM player AS p
    JOIN player_attributes AS pa ON p.player_api_id = pa.player_api_id
    ORDER BY pa.overall_rating DESC
)
```

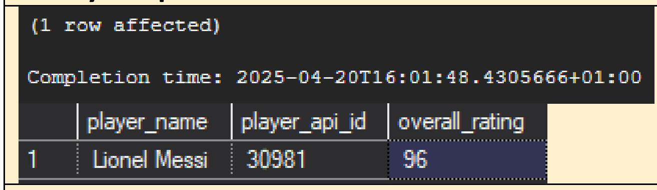
```sql
UPDATE player_attributes
SET overall_rating = overall_rating + 2
WHERE id IN (SELECT player_attribute_id FROM TopPlayer);
```

Query Output Before UPDATE Statement



	player_name	player_api_id	overall_rating
1	Lionel Messi	30981	94

Query Output After UPDATE Statement

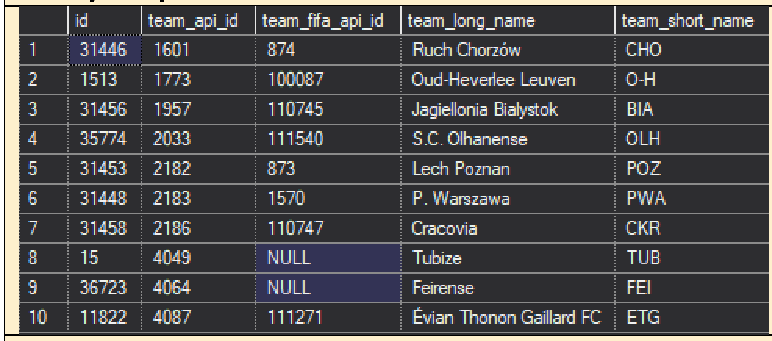


```
(1 row affected)

Completion time: 2025-04-20T16:01:48.4305666+01:00
```

	player_name	player_api_id	overall_rating
1	Lionel Messi	30981	96

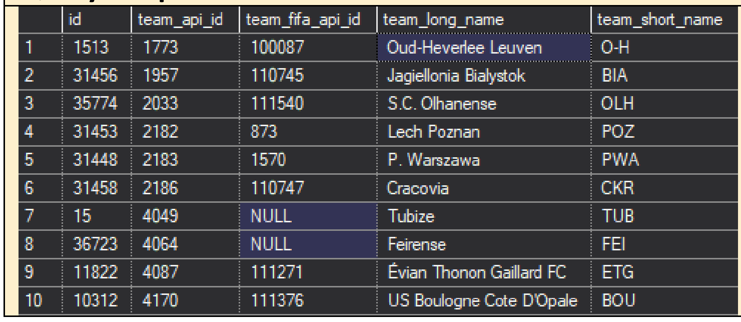
```sql
--Demo E1 Query Two
--Module 7: Modifying and Removing Data
--The query which returns a result I want to update.
SELECT TOP 10 * FROM team
ORDER BY team_api_id

--Description: Deletes the team 'Ruch Chorzów' from the team table.
DELETE FROM team WHERE team_long_name = 'Ruch Chorzów';
```

Query Output Before DELETE Statement



	id	team_api_id	team_fifa_api_id	team_long_name	team_short_name
1	31446	1601	874	Ruch Chorzów	CHO
2	1513	1773	100087	Oud-Heverlee Leuven	O-H
3	31456	1957	110745	Jagiellonia Bialystok	BIA
4	35774	2033	111540	S.C. Olhanense	OLH
5	31453	2182	873	Lech Poznan	POZ
6	31448	2183	1570	P. Warszawa	PWA
7	31458	2186	110747	Cracovia	CKR
8	15	4049	NULL	Tubize	TUB
9	36723	4064	NULL	Feirense	FEI
10	11822	4087	111271	Évian Thonon Gaillard FC	ETG

Query Output After DELETE Statement



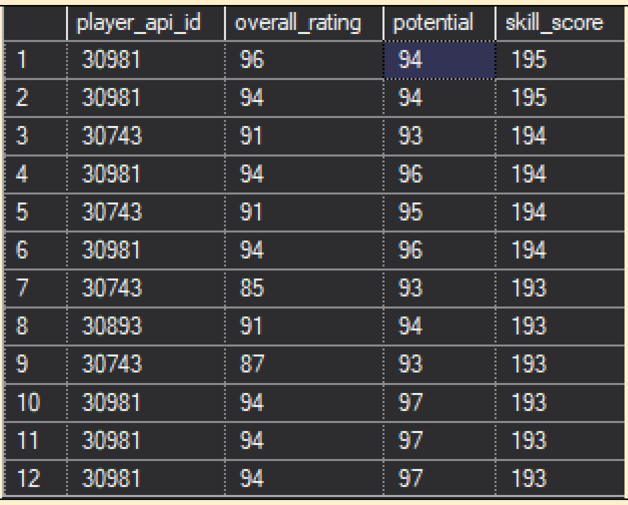
	id	team_api_id	team_fifa_api_id	team_long_name	team_short_name
1	1513	1773	100087	Oud-Heverlee Leuven	O-H
2	31456	1957	110745	Jagiellonia Bialystok	BIA
3	35774	2033	111540	S.C. Olhanense	OLH
4	31453	2182	873	Lech Poznan	POZ
5	31448	2183	1570	P. Warszawa	PWA
6	31458	2186	110747	Cracovia	CKR
7	15	4049	NULL	Tubize	TUB
8	36723	4064	NULL	Feirense	FEI
9	11822	4087	111271	Évian Thonon Gaillard FC	ETG
10	10312	4170	111376	US Boulogne Cote D'Opale	BOU

DEMO 3: Generating Automatic Column Values

TSQL Demo Code Evidence/Results in SSMS

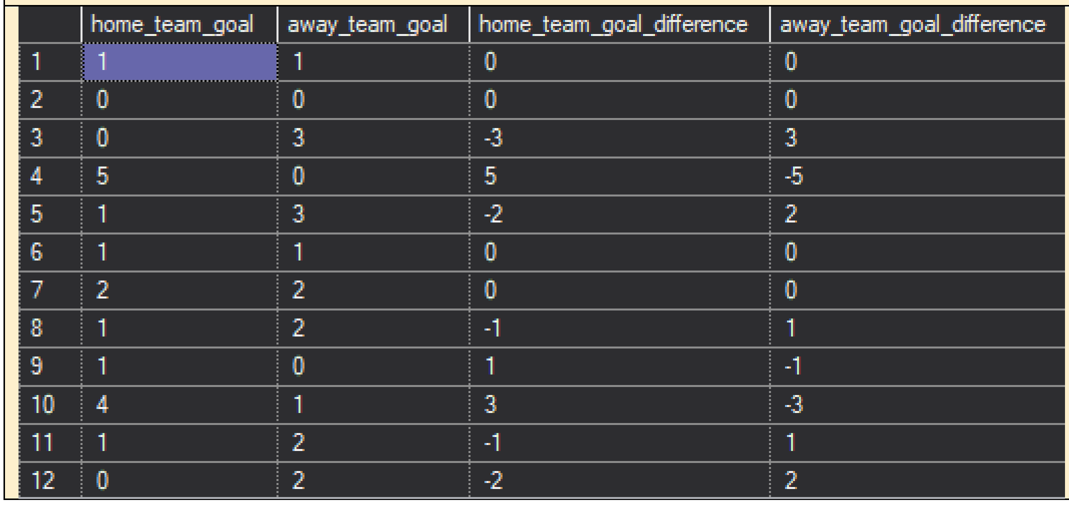
```sql
--Demo E2 Query One
--Module 7: Generating Automatic Column Values
--Description: Adds a new column to the player_attributes table. The values within
the column are automatically generated by performing addition on the dribbling
--and ball control statistics of each player.
ALTER TABLE player_attributes
ADD skill_score AS (dribbling + ball_control);
--Query which selects the api ids, overall ratings, potential ratings and skill
```

```
scores of each player from the player_attributes table. It is ordered by
skill_score in descending order.
SELECT player_api_id, overall_rating, potential, ISNULL(skill_score, '0') AS
skill_score FROM player_attributes
ORDER BY skill_score DESC;
```



	player_api_id	overall_rating	potential	skill_score
1	30981	96	94	195
2	30981	94	94	195
3	30743	91	93	194
4	30981	94	96	194
5	30743	91	95	194
6	30981	94	96	194
7	30743	85	93	193
8	30893	91	94	193
9	30743	87	93	193
10	30981	94	97	193
11	30981	94	97	193
12	30981	94	97	193

```
--Demo E2 Query Two
--Module 7: Generating Automatic Column Values
--Description: Adds two new columns to the match table. One column calculates the
home team goal difference whereas the other calculates the away team goal
difference.
--The values are automatically generated by performing subtraction each way on the
number of home team and away team goals scored within a match.
ALTER TABLE match
ADD home_team_goal_difference AS (home_team_goal - away_team_goal),
       away_team_goal_difference AS (away_team_goal - home_team_goal);
--Query which selects the home team and away team goals scored in each match with
the calculated goal difference for each team in said match.
SELECT home_team_goal, away_team_goal, home_team_goal_difference,
away_team_goal_difference
FROM match;
```



	home_team_goal	away_team_goal	home_team_goal_difference	away_team_goal_difference
1	1	1	0	0
2	0	0	0	0
3	0	3	-3	3
4	5	0	5	-5
5	1	3	-2	2
6	1	1	0	0
7	2	2	0	0
8	1	2	-1	1
9	1	0	1	-1
10	4	1	3	-3
11	1	2	-1	1
12	0	2	-2	2

MODULE 8: USING BUILT-IN FUNCTIONS

Why do programmers use built-in functions?

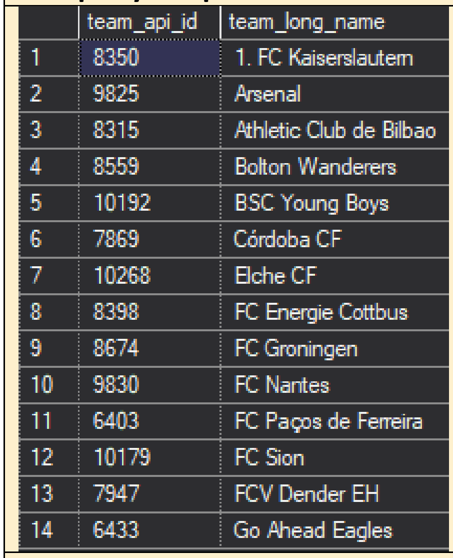
SQL, as well as the vast majority of programming languages, use functions. They are blocks of reusable code that can be repeatedly called upon to perform an instruction or set of instructions. The biggest reason programmers use built-in functions is because it allows complex programs to be decomposed.

DEMO 1: Writing Queries with Built-in Functions

TSQL Demo Code Evidence/Results in SSMS

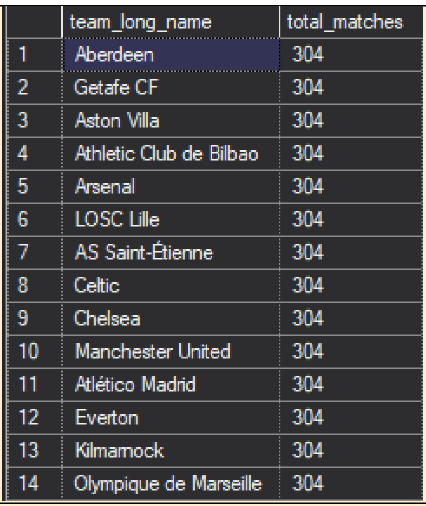
```
--Demo E3 Query One
--Module 8: Writing Queries with Built-In Functions
--Description: The query calculates the total number of matches each team has
played, regardless of whether they are home or away.
--It uses the count built-in function.
WITH teams AS (
    SELECT DISTINCT t.team_api_id, CAST(t.team_long_name AS varchar(MAX)) AS
team_long_name
    FROM team AS t
    JOIN match AS m
    ON t.team_api_id = m.home_team_api_id OR t.team_api_id = m.away_team_api_id
)
SELECT
    t.team_long_name,
    COUNT(m.match_api_id) AS total_matches
FROM teams AS t
JOIN match AS m
       ON t.team_api_id = m.home_team_api_id OR t.team_api_id = m.away_team_api_id
GROUP BY t.team_long_name
ORDER BY total_matches DESC;
```

Subquery Output



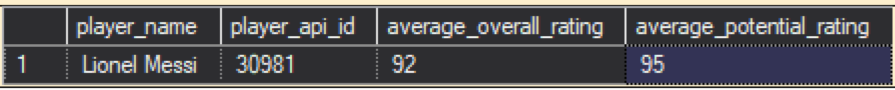
	team_api_id	team_long_name
1	8350	1. FC Kaiserslautern
2	9825	Arsenal
3	8315	Athletic Club de Bilbao
4	8559	Bolton Wanderers
5	10192	BSC Young Boys
6	7869	Córdoba CF
7	10268	Elche CF
8	8398	FC Energie Cottbus
9	8674	FC Groningen
10	9830	FC Nantes
11	6403	FC Paços de Ferreira
12	10179	FC Sion
13	7947	FCV Dender EH
14	6433	Go Ahead Eagles

Entire Query Output



	team_long_name	total_matches
1	Aberdeen	304
2	Getafe CF	304
3	Aston Villa	304
4	Athletic Club de Bilbao	304
5	Arsenal	304
6	LOSC Lille	304
7	AS Saint-Étienne	304
8	Celtic	304
9	Chelsea	304
10	Manchester United	304
11	Atlético Madrid	304
12	Everton	304
13	Kilmarnock	304
14	Olympique de Marseille	304

```
--Demo E3 Query Two
--Module 8: Writing Queries with Built-In Functions
--Description: This query selects the player with the highest average overall and
potential ratings, using the built-in AVG function.
SELECT TOP 1 CAST(p.player_name AS nvarchar(MAX)) AS player_name, p.player_api_id,
AVG(pa.overall_rating) AS average_overall_rating, AVG(pa.potential) AS
average_potential_rating
FROM player AS p
JOIN player_attributes AS pa
ON p.player_api_id = pa.player_api_id
GROUP BY CAST(p.player_name AS nvarchar(MAX)), p.player_api_id
ORDER BY average_overall_rating DESC
```

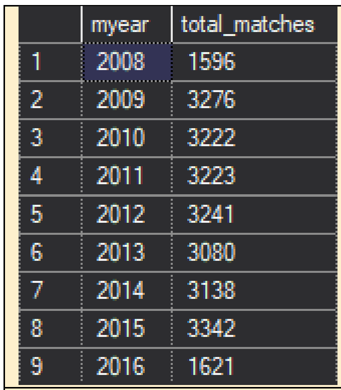


	player_name	player_api_id	average_overall_rating	average_potential_rating
1	Lionel Messi	30981	92	95

DEMO 2: Using Conversion Functions

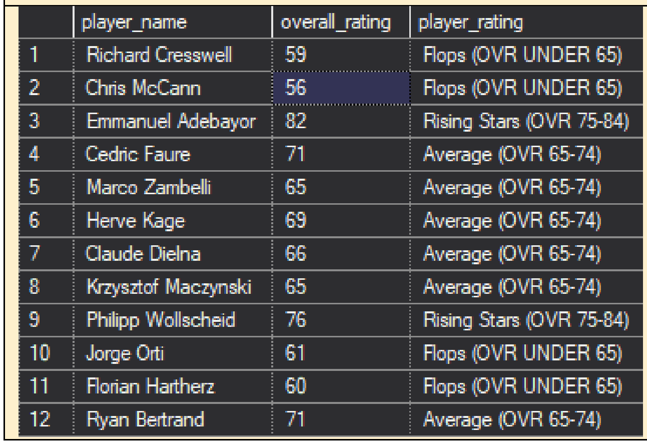
TSQL Demo Code Evidence/Results in SSMS

```
--Demo E4 Query One
--Module 8: Using Conversion Functions
--Description: Counts the total matches played per year in the match table.
--The query extracts just the year part of the mdate and the count function is used
to count all the number of matches for each year.
--The result is grouped and ordered by the myear.
SELECT YEAR(CAST(CAST(mdate AS varchar(MAX)) AS datetime)) AS myear, COUNT(*) AS
total_matches
FROM match
GROUP BY YEAR(CAST(CAST(mdate AS varchar(MAX)) AS datetime))
ORDER BY myear;
```



	myear	total_matches
1	2008	1596
2	2009	3276
3	2010	3222
4	2011	3223
5	2012	3241
6	2013	3080
7	2014	3138
8	2015	3342
9	2016	1621

```
--Demo E4 Query Two
--Module 8: Using Conversion Functions
--Description: This query selects distinct player names and their overall ratings
by joining onto the player_attributes table.
--It uses a case statement to classify the players into rating tiers based on their
overall ratings.
--The query uses a conversion function as it converts the data type of the player
name from text to varchar, so that it can be selected as distinct.
SELECT DISTINCT CAST(p.player_name AS varchar(MAX)) AS player_name,
pa.overall_rating,
       CASE
              WHEN pa.overall_rating >= 90 THEN 'Goats (OVR 90+'
              WHEN pa.overall_rating >= 85 THEN 'Professionals (OVR 85-89)'
              WHEN pa.overall_rating >= 75 THEN 'Rising Stars (OVR 75-84)'
              WHEN pa.overall_rating >= 65 THEN 'Average (OVR 65-74)'
              ELSE 'Flops (OVR UNDER 65)'
       END AS player_rating
FROM player AS p
JOIN player_attributes AS pa ON p.player_api_id = pa.player_api_id;
```



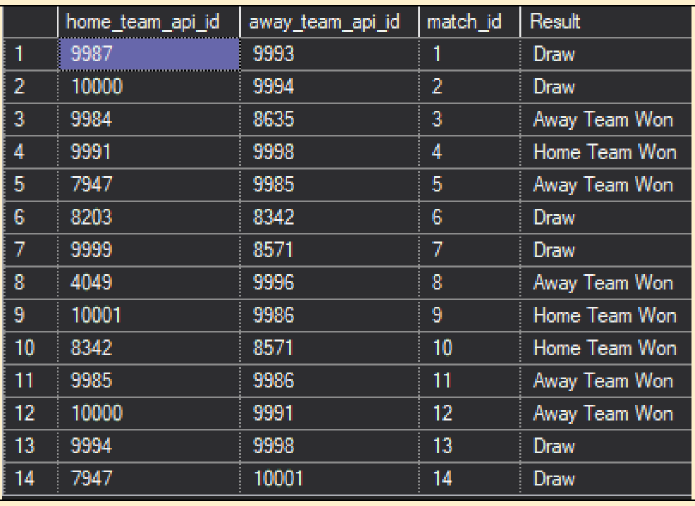
	player_name	overall_rating	player_rating
1	Richard Cresswell	59	Flops (OVR UNDER 65)
2	Chris McCann	56	Flops (OVR UNDER 65)
3	Emmanuel Adebayor	82	Rising Stars (OVR 75-84)
4	Cedric Faure	71	Average (OVR 65-74)
5	Marco Zambelli	65	Average (OVR 65-74)
6	Herve Kage	69	Average (OVR 65-74)
7	Claude Dielna	66	Average (OVR 65-74)
8	Krzysztof Maczynski	65	Average (OVR 65-74)
9	Philipp Wollscheid	76	Rising Stars (OVR 75-84)
10	Jorge Orti	61	Flops (OVR UNDER 65)
11	Florian Hartherz	60	Flops (OVR UNDER 65)
12	Ryan Bertrand	71	Average (OVR 65-74)

DEMO 3: Using Logical Functions

TSQL Demo Code Evidence/Results in SSMS

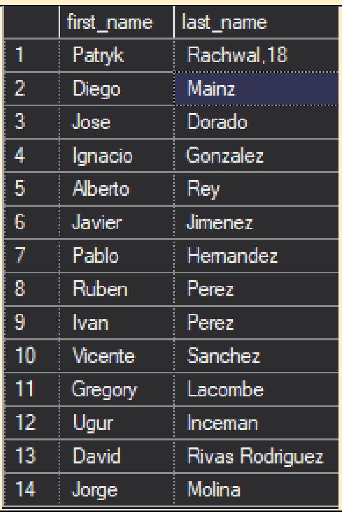
```
--Demo F1 Query One
--Module 8: Using Logical Functions
--Description: The query selects a list of matches from the match table displaying
the ids of the home and away teams,
--the matchIds; creates a result row based on the final score using IIF() logic.
```

```
SELECT home_team_api_id, away_team_api_id, id AS match_id,
       IIF(home_team_goal > away_team_goal, 'Home Team Won',
              IIF(home_team_goal < away_team_goal, 'Away Team Won', 'Draw')
       ) AS Result
FROM match;
```



	home_team_api_id	away_team_api_id	match_id	Result
1	9987	9993	1	Draw
2	10000	9994	2	Draw
3	9984	8635	3	Away Team Won
4	9991	9998	4	Home Team Won
5	7947	9985	5	Away Team Won
6	8203	8342	6	Draw
7	9999	8571	7	Draw
8	4049	9996	8	Away Team Won
9	10001	9986	9	Home Team Won
10	8342	8571	10	Home Team Won
11	9985	9986	11	Away Team Won
12	10000	9991	12	Away Team Won
13	9994	9998	13	Draw
14	7947	10001	14	Draw

```
--Demo F1 Query Two
--Module 8: Using Logical Functions
--Description: The query selects the player names from the player table and splits
them into first and last names using string functions.
SELECT SUBSTRING(CAST(player_name AS varchar(MAX)), 1, CHARINDEX(' ',
CAST(player_name AS varchar(MAX))) - 1) AS first_name,
       SUBSTRING(
              CAST(player_name AS varchar(MAX)),
              CHARINDEX(' ', CAST(player_name AS varchar(MAX))) + 1,
              LEN(CAST(player_name AS varchar(MAX))) - CHARINDEX(' ',
CAST(player_name AS varchar(MAX)))
       ) AS last_name
FROM player
WHERE CHARINDEX(' ', CAST(player_name AS varchar(MAX))) > 0;
```



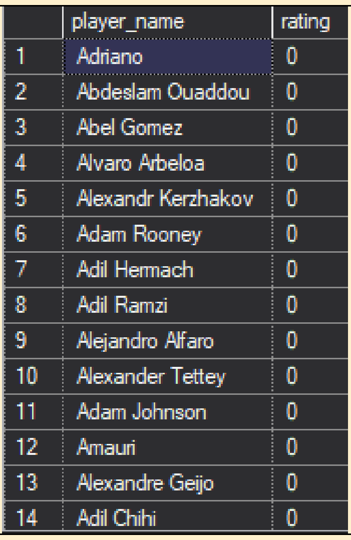
	first_name	last_name
1	Patryk	Rachwal,18
2	Diego	Mainz
3	Jose	Dorado
4	Ignacio	Gonzalez
5	Alberto	Rey
6	Javier	Jimenez
7	Pablo	Hernandez
8	Ruben	Perez
9	Ivan	Perez
10	Vicente	Sanchez
11	Gregory	Lacombe
12	Ugur	Inceman
13	David	Rivas Rodriguez
14	Jorge	Molina

DEMO 4: Using Functions to Work with NULL

TSQL Demo Code Evidence/Results in SSMS

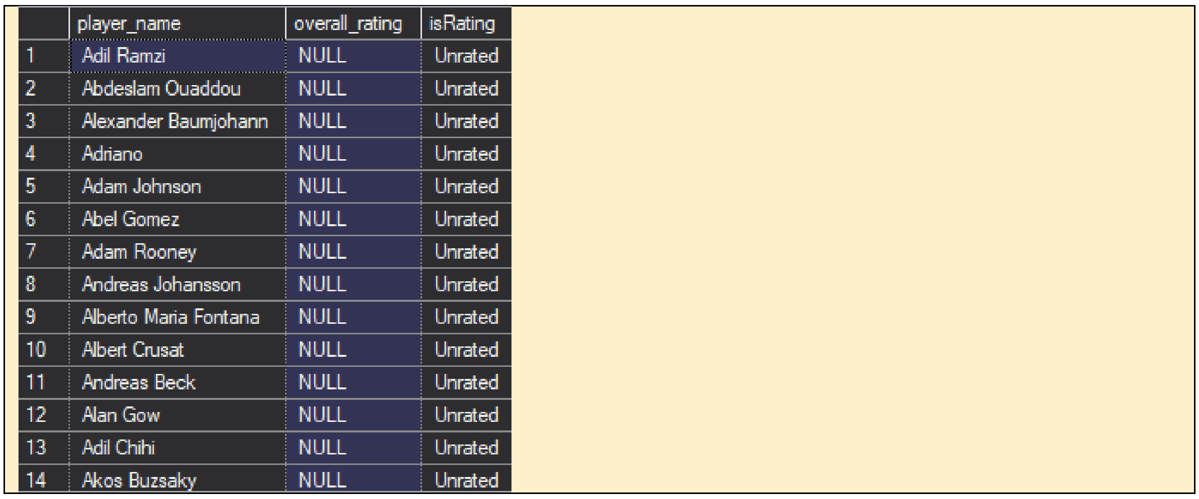
```
--Demo F2 Query One
```

```
--Module 8: Using Functions to work with NULL.
--Description: The query selects distinct player names with a rating value.
--The rating value is assigned by using the COALESCE function. It works with nulls
by checking if overall_rating or if both overall_rating and potential are null.
--If overall_rating is NULL, it falls back to the potential rating, and if both are
NULL, it defaults to 0.
--The results are ordered by the rating in descending order.
SELECT DISTINCT
       CAST(p.player_name AS varchar(MAX)) AS player_name,
       COALESCE(pa.overall_rating, pa.potential, 0) AS rating
FROM player AS p
JOIN player_attributes AS pa
ON p.player_api_id = pa.player_api_id
ORDER BY rating;
```



```
    player_name         rating
1   Adriano             0
2   Abdeslam Ouaddou    0
3   Abel Gomez          0
4   Alvaro Arbeloa      0
5   Alexandr Kerzhakov  0
6   Adam Rooney         0
7   Adil Hermach        0
8   Adil Ramzi          0
9   Alejandro Alfaro    0
10  Alexander Tettey    0
11  Adam Johnson        0
12  Amauri              0
13  Alexandre Geijo     0
14  Adil Chihi          0
```

```
--Demo F2 Query Two
--Module 8: Using Functions to work with NULL.
--Description: The query selects distinct player names and their overall ratings by
joining the player table with the player_attributes table, based on api id.
--The query also labels each player based on whether they have an assigned rating
or whether it is null.
--To do this, it makes use of a case statement inside the select statement which
creates an 'isRating' column.
--It checks if the overall rating of a player is null and assigns an 'unrated'
label to be outputted. Else, a 'rated' label is outputted.
--The result query is ordered by overall rating.
SELECT DISTINCT
       CAST(p.player_name AS varchar(MAX)) AS player_name,
       pa.overall_rating,
       CASE
              WHEN overall_rating IS NULL THEN 'Unrated'
              ELSE 'Rated'
       END AS isRating
FROM player AS p
JOIN player_attributes AS pa
ON p.player_api_id = pa.player_api_id
ORDER BY overall_rating;
```



	player_name	overall_rating	isRating
1	Adil Ramzi	NULL	Unrated
2	Abdeslam Ouaddou	NULL	Unrated
3	Alexander Baumjohann	NULL	Unrated
4	Adriano	NULL	Unrated
5	Adam Johnson	NULL	Unrated
6	Abel Gomez	NULL	Unrated
7	Adam Rooney	NULL	Unrated
8	Andreas Johansson	NULL	Unrated
9	Alberto Maria Fontana	NULL	Unrated
10	Albert Crusat	NULL	Unrated
11	Andreas Beck	NULL	Unrated
12	Alan Gow	NULL	Unrated
13	Adil Chihi	NULL	Unrated
14	Akos Buzsaky	NULL	Unrated