System Designs & Databases ICA

T-SQL Server – TT-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 3](#_Toc193795096)

[Performance rating 3](#_Toc193795097)

[INTRO 3](#_Toc193795098)

[WHY YOU SHOULD LEARN T-SQL 3](#_Toc193795099)

[T-SQL Server Database Overview 3](#_Toc193795100)

[T-SQL Server DATABASE FOR DEMOS 3](#_Toc193795101)

[T-SQL Server DATABASE DIAGRAMS 3](#_Toc193795102)

[T-SQL Supporting Queries 3](#_Toc193795103)

[T-SQL Part One: SQL Server Coding Basics (T-SQL03 to TSQL08) 4](#_Toc193795104)

[MODULE 3: Writing SELECT Queries with single table 4](#_Toc193795105)

[DEMO 1: Writing Simple SELECT query 4](#_Toc193795106)

[DEMO 2: Eliminating Duplicates with DISTINCT 5](#_Toc193795107)

[DEMO 3: Using Column and Table Aliases 7](#_Toc193795108)

[DEMO 4: Writing SIMPLE Case Expressions 7](#_Toc193795109)

[MODULE 4: Joining and querying multiple tables 8](#_Toc193795110)

[DEMO 1: How to provide data from 2 related tables with a Join 8](#_Toc193795111)

[DEMO 2: How to query with inner joins 8](#_Toc193795112)

[DEMO 3: How to query with outer joins 8](#_Toc193795113)

[DEMO 4: How to query with cross joins and self joins 8](#_Toc193795114)

[MODULE 5: Sorting and filtering data 8](#_Toc193795115)

[DEMO 1: How to Sort Data 8](#_Toc193795116)

[DEMO 2: How to Filter Data with Predicates 8](#_Toc193795117)

[DEMO 3: How to Filter Data with TOP and OFFSET-FETCH 9](#_Toc193795118)

[DEMO 4: How to work with Unknown Values 9](#_Toc193795119)

[MODULE 6: Working with data types 9](#_Toc193795120)

[DEMO 1: Working with Data Type examples 9](#_Toc193795121)

[DEMO 2: Working with Character Data 9](#_Toc193795122)

[DEMO 3: Working with Date and Time Data 9](#_Toc193795123)

[MODULE 7: Using DML To modify data 9](#_Toc193795124)

[DEMO 1: Adding Data to Tables 9](#_Toc193795125)

[DEMO 2: Modifying and Removing Data 9](#_Toc193795126)

[DEMO 3: Generating Automatic Column Values 9](#_Toc193795127)

[MODULE 8: Using Built-in functions 9](#_Toc193795128)

[DEMO 1: Writing Queries with Built-in Functions 9](#_Toc193795129)

[DEMO 2: Using Conversion Functions 9](#_Toc193795130)

[DEMO 3: Using Logical Functions 10](#_Toc193795131)

[DEMO 4: Using Funcitons to Work with NULL 10](#_Toc193795132)

# 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

## INTRO

## WHY YOU SHOULD LEARN T-SQL

You should learn SQL because it is the standard language for managing and manipulating databases.

# T-SQL Server Database Overview

## T-SQL Server DATABASE FOR DEMOS

European Top Leagues

## T-SQL Server DATABASE DIAGRAMS

## 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'; |
|  |

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

## MODULE 3: Writing SELECT Queries with single table

### DEMO 1: Writing Simple SELECT query

Two Examples per Demo

|  |
| --- |
| **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; |
| --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; |
|  |

### 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 |
| **Result of the entire query:** |
| --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; |
| ... |

### 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; |
|  |
| --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'; |
|  |

### DEMO 4: Writing SIMPLE Case Expressions

|  |
| --- |
| **TSQL Demo Code Evidence/Results in SSMS** |
| --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; |
|  |
| --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; |
|  |

## MODULE 4: Joining and querying multiple tables

Why use Joining and Querying Multiple Tables?

### 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; |
|  |
| --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. |
|  |
|  |

### 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 |
|  |
| --Demo B2 Query Two  --Module 4: How to query with outer 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; |
|  |

### 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: left 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  LEFT JOIN team\_attributes AS ta  ON t.team\_fifa\_api\_id = ta.team\_fifa\_api\_id; |
|  |
| --Demo B3 Query Two  --Module 4: How to query with outer joins  --User Story: left 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  LEFT JOIN team AS t  ON m.home\_team\_api\_id = t.team\_api\_id OR m.away\_team\_api\_id = t.team\_api\_id; |
|  |

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

## MODULE 5: Sorting and filtering data

### DEMO 1: How to Sort Data

### DEMO 2: How to Filter Data with Predicates

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

### DEMO 4: How to work with Unknown Values

## MODULE 6: Working with data types

### DEMO 1: Working with Data Type examples

### DEMO 2: Working with Character Data

### DEMO 3: Working with Date and Time Data

## MODULE 7: Using DML To modify data

Why use DML to modify data?

### DEMO 1: Adding Data to Tables

### DEMO 2: Modifying and Removing Data

### DEMO 3: Generating Automatic Column Values

## MODULE 8: Using Built-in functions

Why do programmers use built-in functions?

### DEMO 1: Writing Queries with Built-in Functions

### DEMO 2: Using Conversion Functions

### DEMO 3: Using Logical Functions

### DEMO 4: Using Funcitons to Work with NULL