Introduction to SQL

Sample manual - first two chapters

Wise Owl Training

TABLE OF CONTENTS (1 of 5)

1	DESIGNING DATABASES	Page
1.1	The Four Stages of Database Design	7
	Stage 1 – Deciding what to Include	7
	Stage 2 – Dividing Data into Tables	8
	Stage 3 – Choosing a Primary Key for each Table	9
	Stage 4 – Creating Relationships and a Database Diagram	10
1.2	Many-to-Many Relationships	11

2	SQL SERVER MANAGEMENT STUDIO	Page
2.1	Starting to Use Management Studio	12
2.2	Object Explorer	13
	Useful Start-up Options	13

3	CREATING A DATABASE AND TABLES	Page
3.1	Creating the Database	14
	The Files Created	14
3.2	Creating Tables	15
3.3	Setting an Identity Primary Key	16
3.4	Creating Columns	17
	Data Types Explained in this Chapter	17
	Other Data Types in SQL Server	17
3.5	Whole Numbers	18
	Integer Field Types	18
	Logical Field Types	18
3.6	Other Numerical Fields	19
	Decimal and Numeric Field Types	19
	Float and Real Data Types	19
3.7	Character Data Types	20
	Types of Character Storage	20
	Variable Length Data Types	20
	Fixed Length Data Types	20
3.8	Date/Time Data Types	21
	Entering Dates into a Table	21
3.9	Default and Null Values	22
	Allowing Nulls	22
	Default Values	22
3.10	Database Diagrams	23
	Creating a Database Diagram	23
	Creating Relationships	24
	Database Diagram Support Objects Error	24

4	QUERIES	Page
4.1	Basic SELECT Statements	25
	Where to Put your Commas	25
4.2	Creating Queries	26
	Starting a New Query	26
	Choosing the Right Database	26
4.3	Running Queries	27
	Parsing a Query	27
	Executing a Query	27
	Viewing Information on a Query's Execution	28
	Cancelling a Running Query	28
	Redirecting Query Output	28
4.4	Dealing with Errors	29
	Displaying Line Numbers	29
4.5	Using IntelliSense	30
	Refreshing IntelliSense	30
4.6	Multiple SQL Commands	31
4.7	Saving, Opening and Closing Queries	32
	Saving Queries	32
	Opening Queries	32
	Closing Queries	33

5	LAYING OUT QUERIES	Page
5.1	Using Case	34
5.2	Indentation and Word Wrap	35
	Changing Tab Settings	35
	Word Wrap	35
5.3	Comments	36
	Commenting Out Blocks of Code	36
5.4	Colours in SQL	37
	Changing the Default Colours	37
5.5	Auto-formatting SQL	38

© Copyright 2025

TABLE OF CONTENTS (2 of 5)

6	THE SELECT STATEMENT	Page
6.1	SELECT Statement Syntax	39
	Mnemonic for Order of Commands	39
6.2	Qualified Tables and Columns	40
	Dragging Tables/Columns onto a Query	40
	Specifying the DBO Schema and Database	40
6.3	Table Aliases	40
	Reason 1 for Aliases – Easier to Refer to Field Names	41
	Reason 2 for Aliases – Joins	42
	Changing a Table Alias	42
6.4	Column Aliases	43
	Basic Column Aliases	43
	Other Ways to Create Column Aliases	43
	Aliases in WHERE and ORDER BY Clauses	44
6.5	Ordering Rows	45
	Simple Sorting	45
	Sorting by Multiple Columns	45
6.6	Miscellaneous SELECT Tricks	46
	*Selecting All Columns Using **	46
	Selecting Unique Rows	46
	Showing Top and Bottom Rows	47
	Including Ties	47
6.7	Using UNION to Combine Results	48

7	QUERY DESIGNER	Page
7.1	Starting Query Designer	49
	What Query Designer is and does	49
	Starting Query Designer	49
7.2	Using Query Designer	50
	Choosing Tables	50
	Adding/Removing Tables	50
	The Parts of Query Designer	51
	The Non-Existent Results Pane/Execute Button	51
	Working with Columns	52
	Finishing Work in Query Designer	52
7.3	Editing Generated SQL	53
7.4	Advanced Features	54
	Inner and Outer Joins	54
	Grouping	54

8	CRITERIA USING WHERE	Page
8.1	The WHERE Clause	55
	Relational Operators	55
8.2	Criteria with Numbers	56
	Using Comparisons	56
	Finding Numbers in a Given Range	56
8.3	Criteria using Text	57
	Exact Matches	57
	Wildcard Matches using LIKE	57
	Using Special Characters with LIKE	58
	Ranges and Wildcards	58
	Using Relational Operators with Text	59
	Case Sensitivity	59
8.4	Criteria for Dates	60
	Using Dates in Criteria	60
	Using Dates with Wildcards	60
8.5	Combining Criteria	61
8.6	Nulls	62
	An Example of Testing for Nulls	62
	Entering Nulls into a Table	62

9	EXPORTING TO EXCEL	Page
9.1	Copying and Pasting	63
	Copying Column Headers by Default	63
9.2	Exporting Data	64
	Step 1 – Getting the Query	64
	Step 2 – Starting to Export Data	64
	Step 3 – Choosing the Source	64
	Step 4 – Choosing the Destination	65
	Step 5 – Specifying the Data to Export	66
	Step 6 – Specifying how to Export	66
	Step 7 – Finishing the Export	66

© Copyright 2025

TABLE OF CONTENTS (3 of 5)

10	CALCULATIONS	Page
10.1	Creating Calculated Columns	68
	Giving Calculated Columns Aliases	68
	Using Column Aliases in ORDER BY Clauses	68
	Column Aliases Don't Work in WHERE Criteria	69
10.2	Using SQL Functions	70
	Typing an SQL Function	70
	Getting the Full List of Functions	70
10.3	Casting Data Types	71
	The Need for Casting	71
	Data Type Precedence	72
	The CAST Function	73
	The CONVERT Function	73
10.4	Numerical Calculations	74
	Mathematical Symbols and BODMAS	74
	The Modulus Operator (%)	74
	Mathematical Functions	75
	The Importance of Casting Numbers for Calculations	76
	A Short-Cut to Forcing the Right Number Type	76
10.5	Text Calculations	77
	Concatenating Text using the CONCAT Function	77
	Concatenating Text Using the Plus Sign (with Data Conversion)	77
	Functions to Turn Numbers into Text	79
	Functions to Search for and Replace Text	79
	Functions for Extracting Text	80
	Changing the Case of Text	80
	Functions for Trimming Text	81
	Other Text Functions	81
	Worked Example – 1	82
	Worked Example – 2	82
	3	82
10.6	Dealing with Nulls	83
	The ISNULL Function	83
	The COALESCE Function	84
10.7	Testing Conditions using IIF	85

11	THE CASE EXPRESSION	Page
11.1	The Searched Case Expression	86
	Example: Film Bands	86
	Example: Film Era	87
	Using CASE in WHERE Criteria	87
11.2	The Simple Case Statement	88
11.3	Nested CASE Statements	89

12	DATE CALCULATIONS	Page
12.1	How Dates and Times Work	90
	How SQL Server Stores Dates and Times	90
	Displaying Dates/Times	90
	GETDATE – the Current Date/Time	91
	Dates Prefer American Format	91
12.2	Formatting Dates using FORMAT	92
	The Available Codes	92
	Using the Culture Argument	93
	How Slow is the Format Function?	93
12.3	Formatting Dates using CONVERT	94
12.4	Parts of a Date: DATEPART and DATENAME	95
	Displaying a Day Suffix	96
12.5	Getting the Difference between Dates	96
	Subtracting One Date from Another	97
	The DATEDIFF Function	97
12.6	Calculating Ages Correctly	98
	Using DateDiff	98
	Dividing Someone's Age in Days by 365	98
	Getting the Exact Age	99
12.7	Adding Dates using DATEADD	100

© Copyright 2025

TABLE OF CONTENTS (4 of 5)

13	JOINS	Page
13.1	Overview of Joins	101
	What is a Join?	101
	The Types of Join	101
13.2	Understanding your Database	102
	How Relationships Work (Reminder)	102
13.3	Easy Joins, using Query Designer	103
13.4	Inner Joins	104
	The Syntax of an Inner Join	104
	Our Example – Joining the Film and Director Tables	104
	Joining more than One Table	105
	Variations on Inner Join Syntax	105
	Composite Joins	106
	Joining by Expressions	106
13.5	Outer Joins	108
	Outer Joins using Query Designer	108
	Outer Joins in SQL	109
	Left and Right Outer Joins	109
	Picking Out Unmatched Rows	110
	Full Outer Joins	110
13.6	Cross Joins	111
	A Practical Example of Cross Joins	111
13.7	Self-Joins	112

14	SUMMARISING DATA	Page
14.1	Simple Summarising	113
	Syntax of a Simple Summary	113
14.2	Counting	114
	Counting All of a Table's Rows	114
	Counting Non-Null Columns	114
	Counting Unique Values	115
14.3	Grouping	116
	Why you Need GROUP BY	116
	The GROUP BY Clause	116
	Grouping by Multiple Columns	117
	Grouping by Expressions	117
	Grouping without Aggregating	118
14.4	Filtering Results using HAVING	119
14.5	Casting Data for (eg) Averages	120
14.6	Dealing with Nulls	121
	The Default Treatment of Nulls	121
	Forcing SQL to Include Nulls	121
14.7	Additional Options when Grouping	122
	Using ALL to Show Missing Rows	122
	Using CUBE to Show All Combinations	122
	Using GROUPING to Show Levels	123

15	VIEWS	Page
15.1	Why Views are Useful	124
	Use 1: Pre-Joining Tables	124
	Use 2: Virtually Renaming Columns	125
15.2	Views using the Designer	126
	Starting the Designer	126
	Choosing Columns	126
	Sorting and Filtering	127
	Adding Grouping	127
	Executing a View	128
	Saving and Closing Views	129
	Seeing your View in Object Explorer	129
	Running a View	130
	Changing a View	130
15.3	Scripting Views	131
	Creating a New View	131
	Changing an Open View in Script	132
	Changing a View's Script from Object Explorer	132
15.4	Switching between the Designer and Scripting	133

© Copyright 2025

TABLE OF CONTENTS (5 of 5)

16	CTES AND DERIVED TABLES	Page
16.1	Multi-Stage Queries	134
16.2	Derived Tables	135
16.3	Single CTEs (Common Table Expressions)	136
	Syntax of Single CTEs	136
	The CTE for our Example	136
16.4	Multiple CTEs	137
	Syntax of Multiple CTEs	137
	Example of a Multiple CTE	138

17	SUBQUERIES	Page
17.1	Single-Value Subqueries	139
	Example: Showing the Name of the Longest Film	139
17.2	ANY, ALL, IN and NOT IN	140
17.3	Correlated Subqueries	141
	Correlated Subqueries: Definition and Example	141
	Alternatives to Correlated Subqueries	141
	Considering Speed	142
	Using EXISTS to Check whether Rows are Returned	142

18	RANKING AND PERCENTILES	Page
18.1	Ranking and Numbering	143
	Simple Row Numbering	143
18.2	Leading and Lagging	144
	Example of LAG: Actors Born One Week Apart	144
18.3	Percentiles	145
	Percentile Rankings	145

19	WRITING SQL IN AI TOOLS	Page
19.1	Choosing an AI Tool	146
19.2	Blind Queries	147
19.3	Querying a Specific Database	148
	Method 1 – Getting Schema Information	148
	Method 2 – Uploading a Database Diagram	149
	Method 3 – Generating a Schema	150
19.4	Formatting and Refactoring	151
	Comments and Indentation	151
	Rewriting Generated Queries	152
	Turning Views into Readable SQL	153
19.5	Optimising Queries	154
19.6	Debugging SQL	155

© Copyright 2025

CHAPTER 1 - DESIGNING DATABASES

The world runs on relational databases. If you understand the principles upon which these are built, you'll find it much easier to write SQL to get information out of them!

*This manual gives an overview only of database design principles. If you want to delve deeper, try Googling phrases like **Third Normal Form**, **Database Normalisation** or **Entity Diagram**. If nothing else, this will give you an impressive search history in your browser!*

1.1 The Four Stages of Database Design

There are four stages to designing a relational database, shown below (using the example of creating a simple database to hold films; or movies, if you must).

Stage 1 – Deciding what to Include

A good way to do this is to create a spreadsheet of the data you want to include for each film:

Title	Oscars	Director	Date of birth	Studio
Armageddon	0	Michael Bay	17/02/1965	Touchstone Pictures
Bad Boys	0	Michael Bay	17/02/1965	Jerry Bruckheimer Films
Bad Boys II	0	Michael Bay	17/02/1965	Jerry Bruckheimer Films
Dead Poets Society	1	Peter Weir	21/08/1944	Touchstone Pictures
Master and Commander ...	2	Peter Weir	21/08/1944	20th Century Fox
Pearl Harbor	1	Michael Bay	17/02/1965	Touchstone Pictures
The Rock	0	Michael Bay	17/02/1965	Hollywood Pictures
The Truman Show	0	Peter Weir	21/08/1944	Scott Rudin Productions

We want to assign each film to a director, but we don't want to have to type each director's name in over and over again!

The aim of designing a relational database is to ensure that you don't hold information twice:

Title	Oscars	Director
Dead Poets Society	1	Peter Weir
Master and Commander ..	2	Peter Wier
The Truman Show	0	Peter Weird

Not only is holding duplicate information inefficient, but it also means that spelling mistakes will creep in. Here listing out films directed by **Peter Weir** would miss out the last two films, as his name has been misspelt.

© Copyright 2025

Stage 2 – Dividing Data into Tables

Having decided what data you want to include, the next stage of database design is to decide which table each bit of information belongs to:

Title	Oscars	Director	Date of birth	Studio
Armageddon	0	Michael Bay	17/02/1965	Touchstone Pictures
Bad Boys	0	Michael Bay	17/02/1965	Jerry Bruckheimer Films
Bad Boys II	0	Michael Bay	17/02/1965	Jerry Bruckheimer Films
Dead Poets Society	1	Peter Weir	21/08/1944	Touchstone Pictures
Master and Commander ...	2	Peter Weir	21/08/1944	20th Century Fox
Pearl Harbor	1	Michael Bay	17/02/1965	Touchstone Pictures
The Rock	0	Michael Bay	17/02/1965	Hollywood Pictures
The Truman Show	0	Peter Weir	21/08/1944	Scott Rudin Productions

These are all details to do with the film itself.

These are to do with the director (name / birthday).

These are details to do with the studio.

There's no magic wand to make this easier, other than bitter experience of getting it wrong and having to start again! A good guideline is that if you find yourself typing in something twice, it probably belonged in a different table.

For our example above, there are clearly 3 separate entities: films, the directors who made them and the studios which produced them. Here are the fields that each table could contain:

Table	Fields
Film	**Title** and **Oscars Won**, plus something to identify which director and which studio made it
Director	**Director name** and **Date of birth**, plus some unique identifier for the director
Studio	**Studio name**, plus some unique identifier for the studio

What you need to do next is to decide what form these unique identifiers should take.

© Copyright 2025

Stage 3 – Choosing a Primary Key for each Table

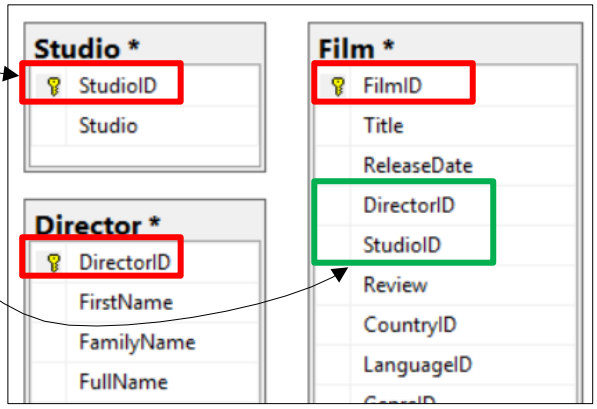
The *primary key* for a table is a field which tells you exactly which record you're considering (for example, if you know a film's **DirectorID** you can look up all of the director's other details).

From the above definition, it follows that two records in a table can't have the same value for the primary key field – the field is unique.

For our example, we could use the director and studio names as our primary keys, but SQL Server works most efficiently if the primary key is as short as possible, so we'll create new fields instead:

The primary key for each table is a number which uniquely identifies which studio, film or director we're looking at.

Each film contains fields specifying who directed the film, and which studio made it.



Here's what **The Sound of Music** would now look like:

FilmId	Title	OscarWins	DirectorID	StudioID
638	The Sound of Music	5	89	4

Including the director's unique number allows us to look up all their other details:

DirectorID	FullName	DoB	Gender
89	Robert Wise	10/09/1914	Male

Including the studio's unique number allows us to look up its name:

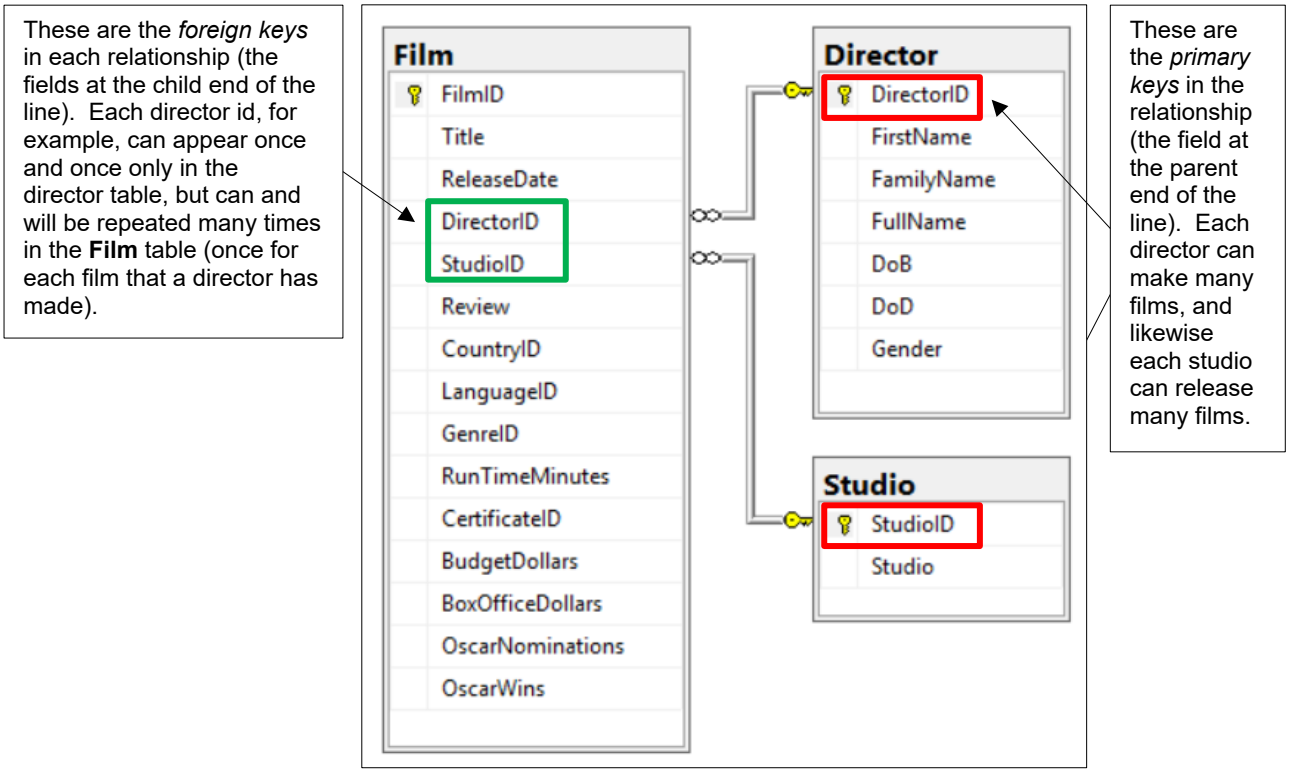
StudioID	Studio
4	20th Century Fox

*If you're beginning to think that relational databases are just like a lot of spreadsheets joined together with a more efficient version of a **VLOOKUP** formula in Excel, you're absolutely right!*

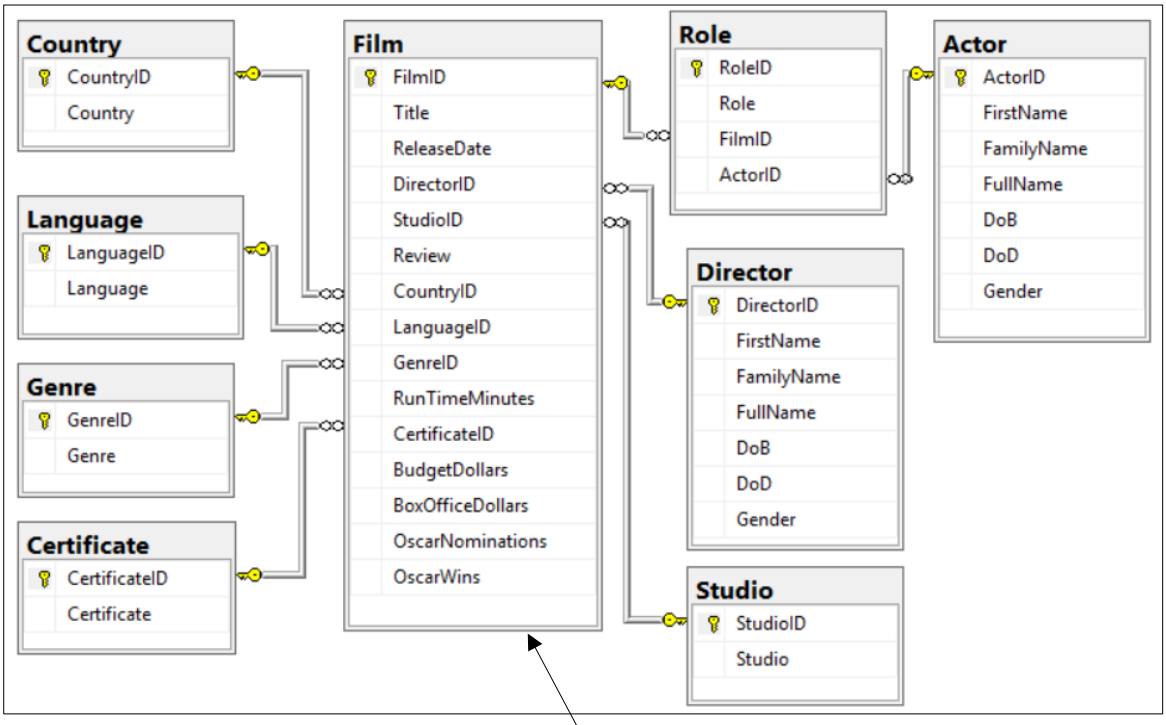
© Copyright 2025

Stage 4 – Creating Relationships and a Database Diagram

The last step in designing a database is to decide for each relationship that you create whether it is *one-to-many* or *many-to-one* (*parent-child* or *child-parent*):



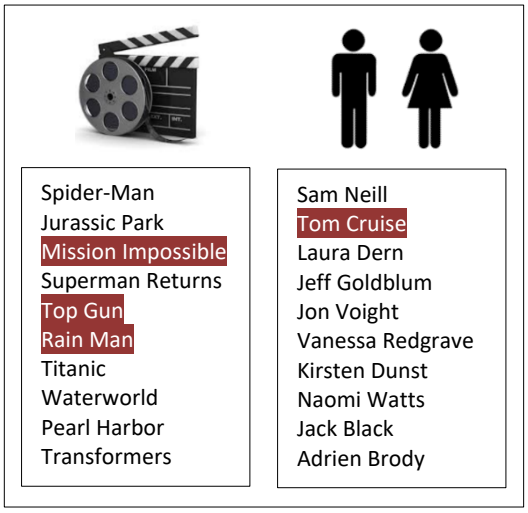
Database diagrams often involve hundreds of tables:



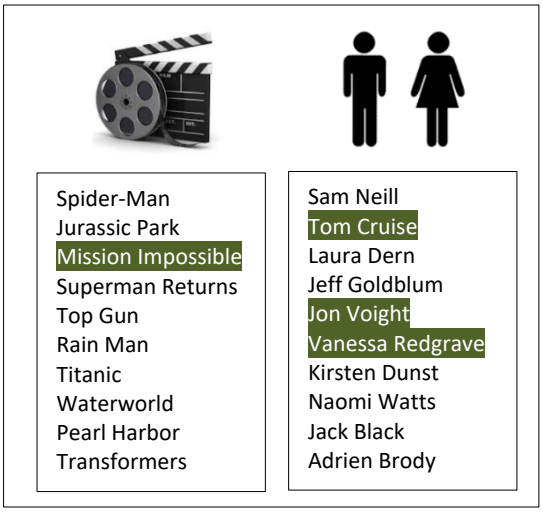
The **Movies** database we'll use in this courseware contains just 9 tables, and hence is untypically simple.

1.2 Many-to-Many Relationships

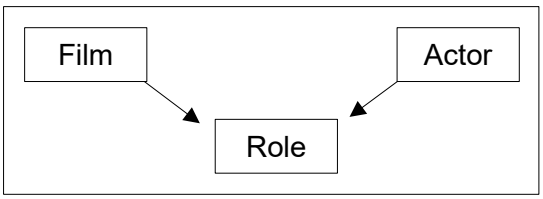
There's no such thing as a many-to-many relationship in SQL Server, but they do exist in real life:



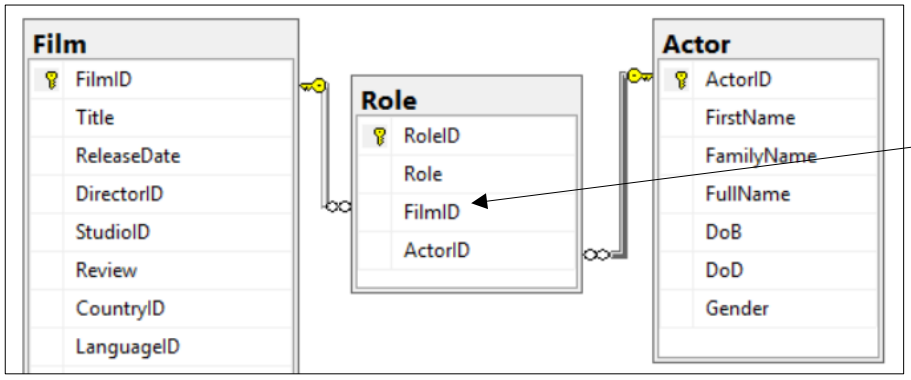
***Tom Cruise** has appeared in lots of films, but equally **Mission: Impossible** has lots of actors in it.*



The solution to this problem is to create a table that is a child to both of the two parent tables, as here:



Here's what the database would look like:



The **Role** table links the **Film** and **Actor** tables. Each film can contain many roles (otherwise a film could only have a single actor), but likewise each actor can have many roles (otherwise they would never work again after completing their first film).

Here are 3 rows from the **Role** table:

RoleID	Role	FilmID	ActorID
1	Ray Ferrier	33	1
2	Dr. Alan Grant	1	2
3	Dr. Ellie Sattler	1	3
202	Nathan Algren	41	1

Film number 1 appears twice in this list, as does actor number 1.

Here are the films and actors who are represented by these rows of data (the duplicate film name was **Jurassic Park**, and duplicated actor turns out to be **Tom Cruise**).

RoleID	Role	FilmID	ActorID	Title	FullName
1	Ray Ferrier	33	1	War of the Worlds	Tom Cruise
2	Dr. Alan Grant	1	2	Jurassic Park	Sam Neill
3	Dr. Ellie Sattler	1	3	Jurassic Park	Laura Dern
202	Nathan Algren	41	1	The Last Samurai	Tom Cruise

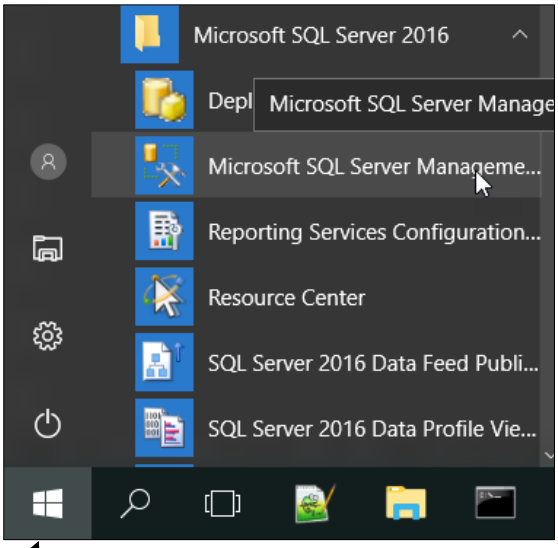
© Copyright 2025

CHAPTER 2 - SQL SERVER MANAGEMENT STUDIO

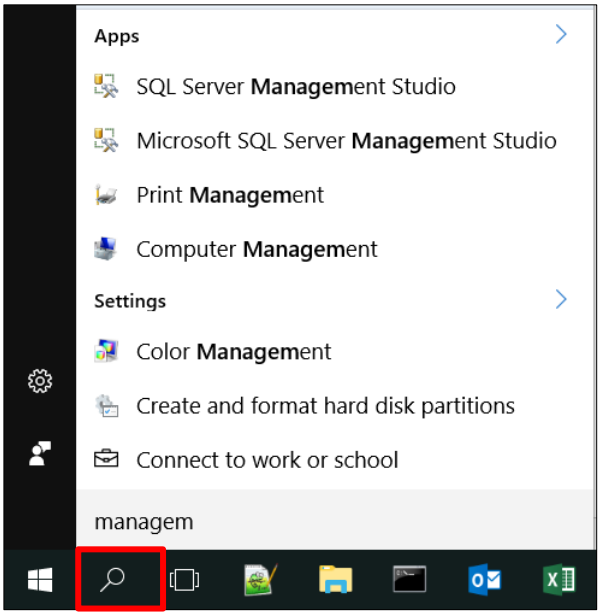
If you're writing SQL to get information out of a database created using SQL Server, the chances are that you'll use *SSMS* (*SQL Server Management Studio*) as your authoring tool.

2.1 Starting to Use Management Studio

You can start SSMS like any other application – here are a couple of ways using Windows 10:



Click on the Windows icon, and choose the program that you want to run …

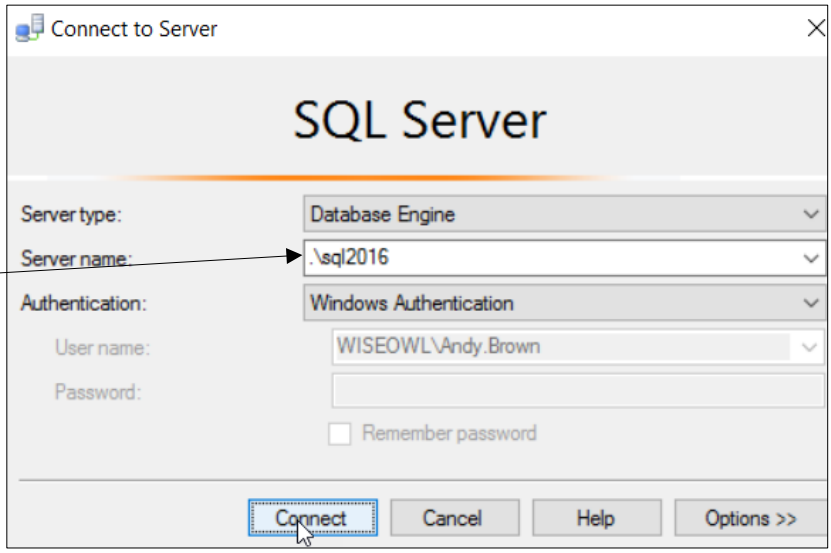


… or click on this tool and type in part of the program name (here typing **managem** has been enough to bring up the program name in the list).

You can then choose a database to use (or *connect to*):

You will see the SQL Server logo on screen while it loads:

You can then choose from the dropdown list which of your company's servers you want to connect to (your IT people should be able to advise on which to choose). If you use Windows authentication, you won't have to type in any more user names or passwords.



© Copyright 2025

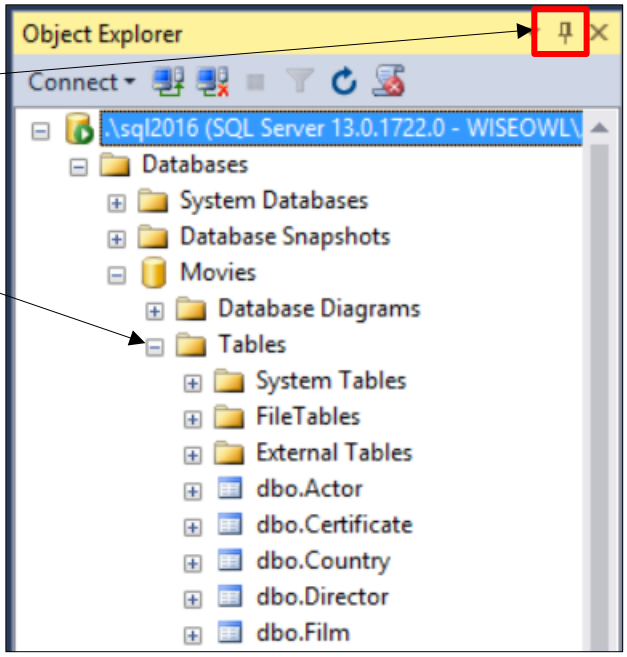
2.2 Object Explorer

When Management Studio loads, you should see the **Object Explorer** window (if it's not visible, press F8 to show it):

If you think Object Explorer is taking up too much room, click on this pin (it will go from vertical to horizontal, and the window will collapse until needed). Click again on the pin to make the window permanently visible again.

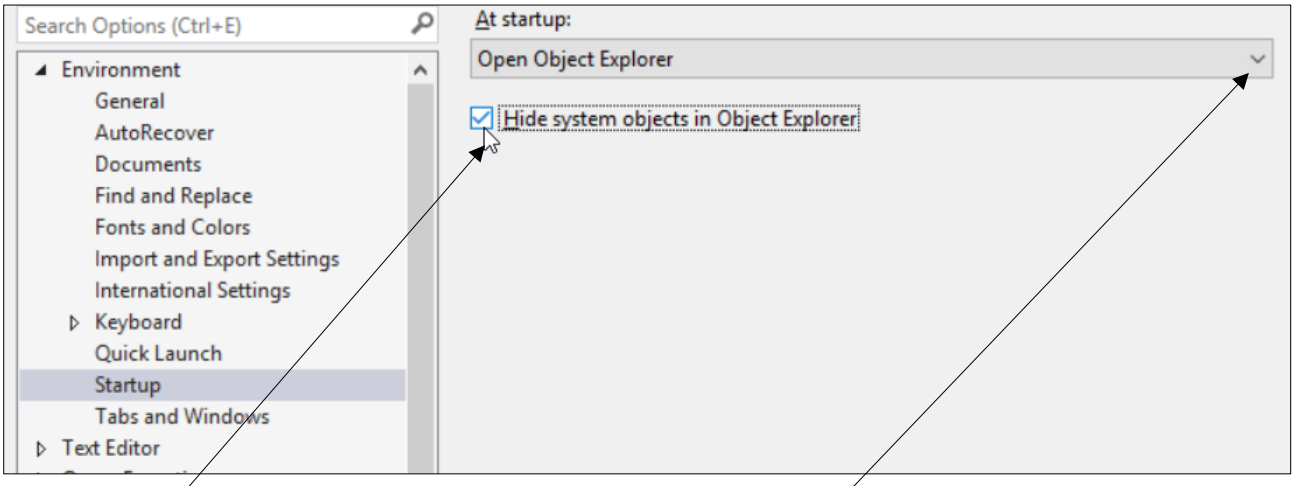
You can click on the ⊞ symbol to expand:

- **Databases**, to see the **Movies** database; then
- **Movies**, to see what it contains; then
- **Tables**, to see what tables there are.



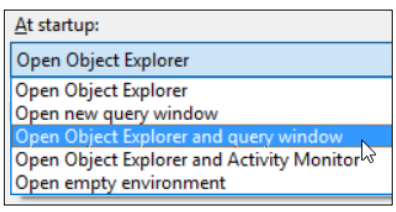
Useful Start-up Options

You can control what happens when you start Management Studio. To do this, from the menu select **Tools ➔ Options**, then complete the dialog box which appears as follows:



a) System objects clutter up SQL Server, and (in this owl's opinion) are best hidden, although you won't see a huge amount of difference.

b) You can click on the drop arrow and choose (for example) to show a blank query as well as Object Explorer whenever you open Management Studio.



© Copyright 2025

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

Wise Owl
Training

What we do!

		Basic training	Advanced training	Systems / consultancy
Office	Microsoft Excel			
	VBA macros			
	Office Scripts			
	Microsoft Access			
Power BI, etc	Power BI and DAX			
	Power Apps			
	Power Automate (both)			
SQL Server	SQL			
	Reporting Services			
	Report Builder			
	Integration Services			
	Analysis Services			
Coding	Visual C#			
	VB programming			
	MySQL			
	Python			

Wise Owl
Training