Systems Design and Databases (CIS1018-N)

Week 5

Querying with Select

Teaching Team

Module Leader & Lecturer: Dr Yar Muhammad
Email: Yar.Muhammad@tees.ac.uk
Office: G0.39 (Greig Building)

Tutor:

- Dr Mengda He
- Mr Mansha Nawaz
- Mr Vishalkumar Thakor

My Academic Hub Time Slots:
Monday 10:00 - 11:00 in Room IT1.13 (Europa Building)
Tuesday 13:00 - 14:00 in Room IT1.13 (Europa Building)

- See Blackboard Ultra for online materials: https://bb.tees.ac.uk/

Lectures & IT Labs

Lectures – Dr Yar Muhammad	Tuesdays @ 2-3 pm	Thursdays @ 1-2 pm
Week 1 – Week 12	CL1.87	

Tutor – Thursday	IT Lab Session Room #: IT2.42
Mr Mansha Nawaz M.Nawaz@tees.ac.uk	Time: 3 – 5 pm

Tutor – Friday	IT Lab Session Room #: OL3
Dr Yar Muhammad Yar.Muhammad@tees.ac.uk	Time: 9 – 11 am & 11 am – 1 pm
Dr Mengda He M.He@tees.ac.uk	Time: 9 – 11 am
Mr Vishalkumar Thakor V.Thakor@tees.ac.uk	Time: 11 am – 1 pm & 1 – 3 pm
Mr Mansha Nawaz M.Nawaz@tees.ac.uk	Time: 1 – 3 pm

Schedule 1/3

Systems Design and Databases CIS1018-N
Weekly Plan for the Activities

Week	Lecturer	Lecture Demo	Lab Exercises & Solutions	ICA Tasks:
01	• Module Introduction, • System Design, • Introduction Databases (DDL, DML, DCL, TCL)	• Requirement List & • MoSCoW Wireframe Design & Templates, • User Stories	• Team Setup, • Hands-on to collect/pick the Requirements from MoSCoW and write Writing User stories on each • **Tutorial 1**	Requirements List & MosCOW, User stories
02	• UML and UML Tool,	• Use Case Diagrams from Requirements List and Wireframe	• Hands-on Use Case Diagrams Activities • **Tutorial 2**	Each Wireframe has associated Use Case Activity Deadline for Team Setup is Week # 2, by Friday 07/10/2022 before 4pm
03	• Sequence Diagrams	• Class Diagrams	• Hands-on Sequence & Class Diagrams Activities • **Tutorial 3**	Each Wireframe has associated Sequence and Class Diagrams
04	• Entity Relationship Diagrams (ERD) A Data Modelling Case Tool for Relational Databases	• Introduction to SQL Server • Walk-through: SQL Quick Guide 1 - How to use SSMS to build Databases	• **Tutorial 4** • Lab Resources: SQL Quick Guide 1	Each Wireframe has associated Class Diagram

Systems Design - UML

Analysis

Design

Schedule 2/3

Databases - TSQL

Week	Lecturer	Lecture Demo	Lab Exercises & Solutions	ICA Tasks:
05	• Querying with Select	**Demo A** – Writing Simple SELECT Statements **Demo B/C** – Eliminating Duplicates with DISTINCT **Demo D** - Writing Simple CASE	• TSQL-Mod03 Lab-Exercise 1-4 • **Tutorial 5**	SQL Task A: TSQL03 Querying with Select • Writing Simple SELECT Statements • Eliminating Duplicates with DISTINCT • Using Column and Table Aliases • Writing Simple CASE Expressions
06	• Querying with Multiple Tables	**Demo B** – Relating 2 or more tables – Joins & Joining multiple tables – inner, outer and cross.	• TSQL-Mod04 Exercise 1-5 • **Tutorial 6**	SQL Task B: TSQL04 – Querying with Multiple Tables • Relating 2 or more tables – Joins • Joining multiple tables – inner, outer and cross.
07	• Sorting and Filtering Data	**Demo A** – Sort with ORDER BY **Demo B** – Filter with WHERE Clause **Demo C** – Filtering with Top OffsetFetch **Demo D** – Handling NULL	• TSQL-Mod05 Exercise 1 – 4 • **Tutorial 7**	SQL Task C: TSQL05 – Sort and Filtering Data • Sort with Order By • Filter with Where By • Filter with top offsetfetch • Handling Nulls
Submission ICA 1 (Group Submission) -> Deadline is Wednesday 16/11/2022 before 4pm				
08	• Working with SQL Server Data	**Demo A** - Conversion in a Query **Demo B** - collation in a query **Demo C** - date and time functions	• TSQL-Mod06 Exercise 1 – 4 • **Tutorial 8**	SQL Task D: TSQL06 – Working with SQL Server Data • Conversion in a Query • collation in a query • date and time functions

Schedule 3/3

Databases - TSQL

09	• Using DML to modify Data	**Demo A** - Adding Data to Tables **Demo B** - Modifying and Removing Data **Demo C** - Generating Automatic Column Values	• TSQL-Mod07 Exercise 1 – 2 • **Tutorial 9**	SQL Task E: TSQL07– Using DML to Modify Data • Adding Data to Tables • Modifying and Removing Data • Generating Automatic Column Values
10	• Using built in Functions	**Demo A** – Scalar Functions **Demo B** – Cast Functions Demo C – If Functions **Demo D** – IsNull Functions	• TSQL-Mod08 Exercise 1 – 3 • **Tutorial 10**	SQL Task F: TSQL08– Using Built-In Functions • Writing Queries with Built-In Functions • Using Conversion Functions • Using Logical Functions • Using Functions to Work with NULL
11	• Walk through SQL Quick Guide 2 - Create a Tables and Relationships via SSMS GUI	• Walk through: • SQL Quick Guide 3 - Create Query, View through Designer	Hands-on: • SQL Server Quick Guide 2	SQL Server – Introduction to SQL Server and SSMS
12	Support	Support	Hands-on: • SQL Server Quick Guide 3	SQL Server – Introduction to SQL Server and SSMS
Submission ICA 2 (Individual Submission) -> Deadline is Wednesday 11/01/2023 before 4pm				

Writing Simple SELECT Statements:

- Elements of the SELECT Statement
- Retrieving Columns from a Table or View
- Displaying Columns
- Using Calculations in the SELECT Clause
- Demonstration: Writing Simple SELECT Statements

Elements of the SELECT Statement

Clause	Expression
SELECT	<select list>
FROM	<table or view>
WHERE	<search condition>
GROUP BY	<group by list>
ORDER BY	<order by list>

Retrieving Columns from a Table or View

- Use SELECT with column list to show columns
- Use FROM to specify the source table or view
  - Specify both schema and object names
- Delimit names if necessary
- End all statements with a semicolon

Keyword	Expression
SELECT	<select list>
FROM	<table or view>

```
SELECT companyname, country
FROM Sales.Customers;
```

Displaying Columns

- Displaying all columns
  - This is not best practice in production code!

```
SELECT *
FROM Sales.Customers;
```

Results | Messages

	custid	companyname	contactname	contacttitle	address	city	region	postalcode	country	phone	fax
1	1	Customer NRZBB	Allen, Michael	Sales Representative	Obere Str. 0123	Berlin	NULL	10092	Germany	030-3456789	030-0123456
2	2	Customer MLTDN	Hassall, Mark	Owner	Avda. de la Constitución 5678	México D.F.	NULL	10077	Mexico	(5) 789-0123	(5) 456-7890
3	3	Customer KBUDE	Peoples, John	Owner	Mataderos 7890	México D.F.	NULL	10097	Mexico	(5) 123-4567	NULL
4	4	Customer HFBZG	Arndt, Torsten	Sales Representative	7890 Hanover Sq.	London	NULL	10046	UK	(171) 456-7890	(171) 456-7891
5	5	Customer HGVLZ	Higginbotham, Tom	Order Administrator	Berguvsvägen 5678	Luleå	NULL	10112	Sweden	0921-67 89 01	0921-23 45 67

- Displaying only specified columns

```
SELECT companyname, country
FROM Sales.Customers;
```

Results | Messages

	companyname	country
1	Customer NRZBB	Germany
2	Customer MLTDN	Mexico
3	Customer KBUDE	Mexico
4	Customer HFBZG	UK
5	Customer HGVLZ	Sweden

Using Calculations in the SELECT Clause

- Calculations are scalar, returning one value per row

Operator	Description
+	Add or concatenate
-	Subtract
*	Multiply
/	Divide
%	Modulo

- Using scalar expressions in the SELECT clause

```
SELECT unitprice, qty, (qty * unitprice)
FROM Sales.OrderDetails;
```

Results | Messages

	unitprice	qty	(No column name)
1	14.00	12	168.00
2	9.80	10	98.00
3	34.80	5	174.00
4	18.60	9	167.40
5	42.40	40	1696.00

Demonstration A - AdventureWorksLT2019: Writing Simple SELECT Statements

In this demonstration you will see how to: Use simple SELECT queries

```
-- Demo Queries are below
USE AdventureWorksLT2019;

SELECT *
FROM SalesLT.ProductCategory;

SELECT ProductCategoryID, ParentProductCategoryID,
Name, rowguid, ModifiedDate
FROM SalesLT.ProductCategory;

SELECT ProductNumber, Name, Color, ListPrice
FROM SalesLT.Product;

SELECT Title, FirstName, LastName, CompanyName,
EmailAddress
FROM SalesLT.customer;


SELECT SalesOrderID, ProductID, UnitPrice,
OrderQty, (UnitPrice * OrderQty)
FROM SalesLT.SalesOrderDetail;
```

Results | Messages

	ProductCategoryID	ParentProductCategoryID	Name	rowguid	ModifiedDate
1	1	NULL	Bikes	CFBDA25C-DF71-47A7-B81B-64EE161AA37C	2002-06-01 00:00:00.000
2	2	NULL	Components	C657828D-D808-4ABA-91A3-AF2CE02300E9	2002-06-01 00:00:00.000
3	3	NULL	Clothing	10A7C342-CA82-48D4-8A38-46A2EB089B74	2002-06-01 00:00:00.000
4	4	NULL	Accessories	2BE3BE36-D9A2-4EEE-B593-ED895D97C2A6	2002-06-01 00:00:00.000
5	5	1	Mountain Bikes	2D364ADE-264A-433C-B092-4FCBF3804E01	2002-06-01 00:00:00.000

Results | Messages

	ProductCategoryID	ParentProductCategoryID	Name	rowguid	ModifiedDate
1	1	NULL	Bikes	CFBDA25C-DF71-47A7-B81B-64EE161AA37C	2002-06-01 00:00:00.000
2	2	NULL	Components	C657828D-D808-4ABA-91A3-AF2CE02300E9	2002-06-01 00:00:00.000
3	3	NULL	Clothing	10A7C342-CA82-48D4-8A38-46A2EB089B74	2002-06-01 00:00:00.000
4	4	NULL	Accessories	2BE3BE36-D9A2-4EEE-B593-ED895D97C2A6	2002-06-01 00:00:00.000
5	5	1	Mountain Bikes	2D364ADE-264A-433C-B092-4FCBF3804E01	2002-06-01 00:00:00.000

Results | Messages

	ProductNumber	Name	Color	ListPrice
1	FR-R92B-58	HL Road Frame - Black, 58	Black	1431.50
2	FR-R92R-58	HL Road Frame - Red, 58	Red	1431.50
3	HL-U509-R	Sport-100 Helmet, Red	Red	34.99
4	HL-U509	Sport-100 Helmet, Black	Black	34.99
5	SO-B909-M	Mountain Bike Socks, M	White	9.50

Results | Messages

	Title	FirstName	LastName	CompanyName	EmailAddress
1	Mr.	Orlando	Gee	A Bike Store	orlando0@adventure-works.com
2	Mr.	Keith	Harris	Progressive Sports	keith0@adventure-works.com
3	Ms.	Donna	Carreras	Advanced Bike Components	donna0@adventure-works.com
4	Ms.	Janet	Gates	Modular Cycle Systems	janet1@adventure-works.com
5	Mr.	Lucy	Harrington	Metropolitan Sports Supply	lucy0@adventure-works.com

Results | Messages

	SalesOrderID	ProductID	UnitPrice	OrderQty	(No column name)
1	71774	836	356.898	1	356.898
2	71774	822	356.898	1	356.898
3	71776	907	63.90	1	63.90
4	71780	905	218.454	4	873.816
5	71780	983	461.694	2	923.388

Demonstration A with TSQL: Writing Simple SELECT Statements

In this demonstration you will see how to: Use simple SELECT queries

```
-- Demo Queries are below

USE TSQL;

SELECT *
FROM Production.Categories;

SELECT CategoryID, CategoryName, Description
FROM Production.Categories;

SELECT ProductID, ProductName, UnitPrice, supplierid,
categoryid FROM Production.Products;

SELECT custid, companyname, contactName, city, phone
FROM Sales.customers;

SELECT ProductID, productName, unitPrice, (unitPrice *
1.1) FROM Production.Products;


SELECT OrderID, ProductID, UnitPrice, Qty, (UnitPrice
* Qty)
FROM Sales.OrderDetails;
```

Results | Messages

	categoryid	categoryname	description
1	1	Beverages	Soft drinks, coffees, teas, beers, and ales
2	2	Condiments	Sweet and savory sauces, relishes, spreads, and ...
3	3	Confections	Desserts, candies, and sweet breads
4	4	Dairy Products	Cheeses
5	5	Grains/Cereals	Breads, crackers, pasta, and cereal

Results | Messages

	CategoryID	CategoryName	Description
1	1	Beverages	Soft drinks, coffees, teas, beers, and ales
2	2	Condiments	Sweet and savory sauces, relishes, spreads, and ...
3	3	Confections	Desserts, candies, and sweet breads
4	4	Dairy Products	Cheeses
5	5	Grains/Cereals	Breads, crackers, pasta, and cereal

Results | Messages

	ProductID	ProductName	UnitPrice	supplierid	categoryid
1	1	Product HHYDP	18.00	1	1
2	2	Product RECZE	19.00	1	1
3	3	Product IMEHJ	10.00	1	2
4	4	Product KSBRM	22.00	2	2
5	5	Product EPEIM	21.35	2	2

Results | Messages

	custid	companyname	contactName	city	phone
1	1	Customer NRZBB	Allen, Michael	Berlin	030-3456789
2	2	Customer MLTDN	Hassall, Mark	México D.F.	(5) 789-0123
3	3	Customer KBUDE	Peoples, John	México D.F.	(5) 123-4567
4	4	Customer HFBZG	Arndt, Torsten	London	(171) 456-7890
5	5	Customer HGVLZ	Higginbotham, Tom	Luleå	0921-67 89 01

Results | Messages

	ProductID	productName	unitPrice	(No column name)
1	1	Product HHYDP	18.00	19.80000
2	2	Product RECZE	19.00	20.90000
3	3	Product IMEHJ	10.00	11.00000
4	4	Product KSBRM	22.00	24.20000
5	5	Product EPEIM	21.35	23.48500

Results | Messages

	OrderID	ProductID	UnitPrice	Qty	(No column name)
1	10248	11	14.00	12	168.00
2	10248	42	9.80	10	98.00
3	10248	72	34.80	5	174.00
4	10249	14	18.60	9	167.40
5	10249	51	42.40	40	1696.00

SQL Sets and Duplicate Rows

- SQL query results are not truly relational:
  - Rows are not guaranteed to be unique
  - No guaranteed order
- Even unique rows in a source table can return duplicate values for some columns

```
SELECT country
FROM Sales.Customers;
```

```
country
-----------
Argentina
Argentina
Belgium
Austria
Austria
```

Results | Messages

	country
1	Germany
2	Mexico
3	Mexico
4	UK
5	Sweden
6	Germany
7	France
8	Spain
9	France
10	Canada
11	UK
12	Argentina
13	Mexico
14	Switzerland
15	Brazil
16	UK
17	Germany
18	France
19	UK
20	Austria
21	Brazil
22	Spain

Understanding DISTINCT

- DISTINCT specifies that only unique rows can appear in the result set
- Removes duplicates based on column list results, not source table
- Provides uniqueness across set of selected columns
- Removes rows already operated on by WHERE, HAVING, and GROUP BY clauses
- Some queries may improve performance by filtering out duplicates before execution of SELECT clause

SELECT DISTINCT Syntax

```
SELECT DISTINCT <column list>

FROM <table or view>
```

```
SELECT DISTINCT companyname, country
FROM Sales.Customers;
```

```
companyname          country
--------------       -------
Customer AHPOP       UK
Customer AHXHT       Mexico
Customer AZJED       Germany
Customer BSVAR       France
Customer CCFIZ       Poland
```

Results | Messages

	companyname	country
1	Customer AHPOP	UK
2	Customer AHXHT	Mexico
3	Customer AZJED	Germany
4	Customer BSVAR	France
5	Customer CCFIZ	Poland
6	Customer CCKOT	Switzerland
7	Customer CQRAA	Italy
8	Customer CYZTN	Sweden
9	Customer DTDMN	Spain
10	Customer DVFMB	USA
11	Customer EEALV	Canada
12	Customer EFFTC	France
13	Customer ENQZT	France
14	Customer EYHKM	USA
15	Customer FAPSM	Germany
16	Customer FEVNN	Germany
17	Customer FRXZL	Ireland
18	Customer FVXPQ	Venezuela
19	Customer GCJSG	UK
20	Customer GLLAG	Germany
21	Customer GYBBY	UK
22	Customer HFBZG	UK

Demonstration: Eliminating Duplicates with DISTINCT

- SQL Sets and Duplicate Rows
- Understanding DISTINCT
- SELECT DISTINCT Syntax
- Demonstration: Eliminating Duplicates with DISTINCT

- In this demonstration, you will see how to eliminate duplicate rows
- Demo files are below:

TSQL-Mod03-Demonstration B - AdventureWorksLT2019.sql

TSQL-Mod03-Demonstration B - TSQL.sql

TSQL-Mod03-Demonstration C - AWLT2019.sql

Demonstration B with AdventureWorksLT2019: Writing Simple SELECT Statements

```sql
-- Demo Queries are below
USE AdventureWorksLT2019;

SELECT Color, Size
FROM SalesLT.Product;

SELECT DISTINCT Color, Size
FROM SalesLT.Product;

SELECT DISTINCT Color
FROM SalesLT.Product;

SELECT DISTINCT Size
FROM SalesLT.Product;

SELECT DISTINCT ProductID, Color, Size
FROM SalesLT.Product;
```

Results | Messages

	Color	Size
1	Black	58
2	Red	58
3	Red	NULL
4	Black	NULL
5	White	M

Results | Messages

	Color	Size
1	NULL	NULL
2	Black	NULL
3	Black	38
4	Black	40
5	Black	42

Results | Messages

	Color
1	NULL
2	Black
3	Blue
4	Grey
5	Multi

Results | Messages

	Size
1	NULL
2	38
3	40
4	42
5	44

Results | Messages

	ProductID	Color	Size
1	680	Black	58
2	706	Red	58
3	707	Red	NULL
4	708	Black	NULL
5	709	White	M
6	710	White	L
7	711	Blue	NULL
8	712	Multi	NULL
9	713	Multi	S
10	714	Multi	M
11	715	Multi	L
12	716	Multi	XL
13	717	Red	62
14	718	Red	44
15	719	Red	48
16	720	Red	52
17	721	Red	56
18	722	Black	58
19	723	Black	60
20	724	Black	62
21	725	Red	44
22	726	Red	48
23	727	Red	52
24	728	Red	58
25	729	Red	60
26	730	Red	62
27	731	Red	44
28	732	Red	48
29	733	Red	52
30	734	Red	58
31	735	Red	60
32	736	Black	44
33	737	Black	48
34	738	Black	52
35	739	Silver	42

Demonstration B with TSQL: Writing Simple SELECT Statements

```sql
-- Demo Queries are below
USE TSQL;

SELECT categoryid
FROM Production.Products;

SELECT DISTINCT categoryid
FROM Production.Products;

SELECT DISTINCT supplierID
FROM Production.Products;
```

Results

	categoryid
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1
11	1
12	1
13	2
14	2
15	2
16	2
17	2
18	2
19	2
20	2
21	2
22	2

Results

	categoryid
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8

Results

	supplierID
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
11	11
12	12
13	13
14	14
15	15
16	16
17	17
18	18
19	19
20	20
21	21
22	22

Demonstration C with AdventureWorksLT2019: Writing Simple SELECT Statements

```sql
-- Demo Queries are below
USE AdventureWorksLT2019;

SELECT Color, Size
FROM SalesLT.Product;

SELECT DISTINCT Color, Size
FROM SalesLT.Product;

SELECT DISTINCT Color
FROM SalesLT.Product;

SELECT DISTINCT Size
FROM SalesLT.Product;

SELECT DISTINCT ProductID, Color, Size
FROM SalesLT.Product;
```

Results | Messages

	Color	Size
1	Black	58
2	Red	58
3	Red	NULL
4	Black	NULL
5	White	M
6	White	L
7	Blue	NULL
8	Multi	NULL
9	Multi	S
10	Multi	M

Results | Messages

	Color	Size
1	NULL	NULL
2	Black	NULL
3	Black	38
4	Black	40
5	Black	42
6	Black	44
7	Black	46
8	Black	48
9	Black	52
10	Black	58

Results | Messages

	Color
1	NULL
2	Black
3	Blue
4	Grey
5	Multi
6	Red
7	Silver
8	Silver/Black
9	White
10	Yellow

Results | Messages

	Size
1	NULL
2	38
3	40
4	42
5	44
6	46
7	48
8	50
9	52
10	54
11	56
12	58
13	60
14	62
15	70
16	L
17	M
18	S
19	XL

Results | Messages

	ProductID	Color	Size
1	680	Black	58
2	706	Red	58
3	707	Red	NULL
4	708	Black	NULL
5	709	White	M
6	710	White	L
7	711	Blue	NULL
8	712	Multi	NULL
9	713	Multi	S
10	714	Multi	M
11	715	Multi	L
12	716	Multi	XL
13	717	Red	62
14	718	Red	44
15	719	Red	48
16	720	Red	52
17	721	Red	56
18	722	Black	58
19	723	Black	60
20	724	Black	62
21	725	Red	44
22	726	Red	48

Using Column and Table Aliases

- Use Aliases to Refer to Columns
- Use Aliases to Refer to Tables
- The Impact of Logical Processing Order on Aliases
- Demonstration: Using Column and Table Aliases

Use Aliases to Refer to Columns

- Column aliases using AS

```
SELECT orderid, unitprice, qty AS quantity
FROM Sales.OrderDetails;
```

Results | Messages

	orderid	unitprice	quantity
1	10248	14.00	12
2	10248	9.80	10
3	10248	34.80	5
4	10249	18.60	9
5	10249	42.40	40

- Column aliases using =

```
SELECT orderid, unitprice, quantity = qty
FROM Sales.OrderDetails;
```

Results | Messages

	orderid	unitprice	quantity
1	10248	14.00	12
2	10248	9.80	10
3	10248	34.80	5
4	10249	18.60	9
5	10249	42.40	40

- Accidental column aliases

```
SELECT orderid, unitprice quantity
FROM Sales.OrderDetails;
```

Results | Messages

	orderid	quantity
1	10248	14.00
2	10248	9.80
3	10248	34.80
4	10249	18.60
5	10249	42.40

Use Aliases to Refer to Tables

- Create table aliases in the FROM clause
- Create table aliases with AS

```
SELECT custid, orderdate
FROM Sales.Orders AS SO;
```

- Create table aliases without AS

```
SELECT custid, orderdate
FROM Sales.Orders SO;
```

- Using table aliases in the SELECT clause

```
SELECT SO.custid, SO.orderdate
FROM Sales.Orders AS SO
```

Results | Messages

	custid	orderdate
1	85	2006-07-04 00:00:00.000
2	79	2006-07-05 00:00:00.000
3	34	2006-07-08 00:00:00.000
4	84	2006-07-08 00:00:00.000
5	76	2006-07-09 00:00:00.000
6	34	2006-07-10 00:00:00.000
7	14	2006-07-11 00:00:00.000
8	68	2006-07-12 00:00:00.000
9	88	2006-07-15 00:00:00.000
10	35	2006-07-16 00:00:00.000
11	20	2006-07-17 00:00:00.000
12	13	2006-07-18 00:00:00.000
13	56	2006-07-19 00:00:00.000
14	61	2006-07-19 00:00:00.000
15	65	2006-07-22 00:00:00.000
16	20	2006-07-23 00:00:00.000
17	24	2006-07-24 00:00:00.000
18	7	2006-07-25 00:00:00.000
19	87	2006-07-26 00:00:00.000
20	25	2006-07-29 00:00:00.000
21	33	2006-07-30 00:00:00.000
22	89	2006-07-31 00:00:00.000

The Impact of Logical Processing Order on Aliases

- FROM, WHERE, and HAVING clauses processed before SELECT
- Aliases created in SELECT clause only visible to ORDER BY
- Expressions aliased in SELECT clause may be repeated elsewhere in query

Using Column and Table Aliases

In this demonstration, you will see how to Use column and table aliases

Demonstration A with AdventureWorksLT2019: Writing Simple SELECT Statements

In this demonstration, you will see how to Use column and table Aliases

```
-- Demo Queries are below
USE AdventureWorksLT2019;


SELECT ProductID, Name, ListPrice, (ListPrice * 1.1) as NewListPrice
FROM SalesLT.Product;

SELECT custid, orderdate
FROM Sales.Orders SO;
```

Results | Messages

	ProductID	Name	ListPrice	NewListPrice
1	680	HL Road Frame - Black, 58	1431.50	1574.65000
2	706	HL Road Frame - Red, 58	1431.50	1574.65000
3	707	Sport-100 Helmet, Red	34.99	38.48900
4	708	Sport-100 Helmet, Black	34.99	38.48900
5	709	Mountain Bike Socks, M	9.50	10.45000

Results | Messages

	custid	orderdate
1	85	2006-07-04 00:00:00.000
2	79	2006-07-05 00:00:00.000
3	34	2006-07-08 00:00:00.000
4	84	2006-07-08 00:00:00.000
5	76	2006-07-09 00:00:00.000
6	34	2006-07-10 00:00:00.000
7	14	2006-07-11 00:00:00.000
8	68	2006-07-12 00:00:00.000
9	88	2006-07-15 00:00:00.000
10	35	2006-07-16 00:00:00.000
11	20	2006-07-17 00:00:00.000
12	13	2006-07-18 00:00:00.000
13	56	2006-07-19 00:00:00.000
14	61	2006-07-19 00:00:00.000
15	65	2006-07-22 00:00:00.000
16	20	2006-07-23 00:00:00.000
17	24	2006-07-24 00:00:00.000
18	7	2006-07-25 00:00:00.000
19	87	2006-07-26 00:00:00.000
20	25	2006-07-29 00:00:00.000
21	33	2006-07-30 00:00:00.000
22	89	2006-07-31 00:00:00.000

Demonstration A with TSQL: Writing Simple SELECT Statements

In this demonstration, you will see how to Use column a

```
-- Demo Queries are below
USE TSQL;

SELECT ProductID, productName, unitPrice, (unitPrice * 1.1) as NewListPrice
FROM Production.Products;


SELECT OrderID, ProductID, UnitPrice, Qty, (UnitPrice * Qty) AS [sub-total]
FROM Sales.OrderDetails;


SELECT SO.custid, SO.orderdate
FROM Sales.Orders AS SO
```

Results | Messages

	ProductID	productName	unitPrice	NewListPrice
1	1	Product HHYDP	18.00	19.80000
2	2	Product RECZE	19.00	20.90000
3	3	Product IMEHJ	10.00	11.00000
4	4	Product KSBRM	22.00	24.20000
5	5	Product EPEIM	21.35	23.48500

Results | Messages

	OrderID	ProductID	UnitPrice	Qty	sub-total
1	10248	11	14.00	12	168.00
2	10248	42	9.80	10	98.00
3	10248	72	34.80	5	174.00
4	10249	14	18.60	9	167.40
5	10249	51	42.40	40	1696.00

Results | Messages

	custid	orderdate
1	85	2006-07-04 00:00:00.000
2	79	2006-07-05 00:00:00.000
3	34	2006-07-08 00:00:00.000
4	84	2006-07-08 00:00:00.000
5	76	2006-07-09 00:00:00.000
6	34	2006-07-10 00:00:00.000
7	14	2006-07-11 00:00:00.000
8	68	2006-07-12 00:00:00.000
9	88	2006-07-15 00:00:00.000
10	35	2006-07-16 00:00:00.000
11	20	2006-07-17 00:00:00.000
12	13	2006-07-18 00:00:00.000
13	56	2006-07-19 00:00:00.000
14	61	2006-07-19 00:00:00.000
15	65	2006-07-22 00:00:00.000
16	20	2006-07-23 00:00:00.000
17	24	2006-07-24 00:00:00.000
18	7	2006-07-25 00:00:00.000
19	87	2006-07-26 00:00:00.000
20	25	2006-07-29 00:00:00.000
21	33	2006-07-30 00:00:00.000
22	89	2006-07-31 00:00:00.000

Writing Simple CASE Expressions

- Using CASE Expressions in SELECT Clauses
- Forms of CASE Expressions
- Demonstration: Simple CASE Expressions

Case Expression

- The CASE expression goes through conditions and returns a value when the first condition is met (like an if-then-else statement).
- So, once a condition is true, it will stop reading and return the result.
- If no conditions are true, it returns the value in the ELSE clause.
- If there is no ELSE part and no conditions are true, it returns NULL.

- CASE Syntax

```
CASE
    WHEN condition1 THEN result1
    WHEN condition2 THEN result2
    WHEN conditionN THEN resultN
    ELSE result
END;
```

Using CASE Expressions in SELECT Clauses

- T-SQL CASE expressions return a single (scalar) value
- CASE expressions may be used in:
  - SELECT column list
  - WHERE or HAVING clauses
  - ORDER BY clause
- CASE returns result of expression
  - Not a control-of-flow mechanism
- In SELECT clause, CASE behaves as calculated column requiring an alias

Forms of CASE Expressions

- Two forms of T-SQL CASE expressions are:
- Simple CASE
  - Compares one value to a list of possible values
  - Returns first match
  - If no match, returns value found in optional ELSE clause
  - If no match and no ELSE, returns NULL
- Searched CASE
  - Evaluates a set of predicates, or logical expressions
  - Returns value found in THEN clause matching first expression that evaluates to TRUE

Demonstration D with AdventureWorksLT2019: Writing Simple SELECT Statements

In this demonstration, you will see how to Use column and table Aliases

```sql
-- Demo Queries are below
USE AdventureWorksLT2019;

SELECT ProductNumber, Name, ListPrice, size,
CASE Size
WHEN 'S' THEN 'Small'
WHEN 'M' THEN 'Medium'
WHEN 'L' THEN 'Large'
WHEN 'XL' THEN 'Extra Large'
ELSE Size
END AS SizeLong
FROM SalesLT.Product;

SELECT ProductNumber, Name,ProductCategoryID,
CASE ProductCategoryID
WHEN 5 THEN 'Mountain Bikes'
WHEN 6 THEN 'Road Bikes'
WHEN 7 THEN 'Touring Bikes'
ELSE 'Bike Accessories'
END AS Category
FROM SalesLT.Product
Where productcategoryid in (5, 6, 7)
```

Results | Messages

	ProductNumber	Name	ListPrice	size	SizeLong
1	FR-R92B-58	HL Road Frame - Black, 58	1431.50	58	58
2	FR-R92R-58	HL Road Frame - Red, 58	1431.50	58	58
3	HL-U509-R	Sport-100 Helmet, Red	34.99	NULL	NULL
4	HL-U509	Sport-100 Helmet, Black	34.99	NULL	NULL
5	SO-B909-M	Mountain Bike Socks, M	9.50	M	Medium
6	SO-B909-L	Mountain Bike Socks, L	9.50	L	Large
7	HL-U509-B	Sport-100 Helmet, Blue	34.99	NULL	NULL
8	CA-1098	AWC Logo Cap	8.99	NULL	NULL
9	LJ-0192-S	Long-Sleeve Logo Jersey, S	49.99	S	Small
10	LJ-0192-M	Long-Sleeve Logo Jersey, M	49.99	M	Medium
11	LJ-0192-L	Long-Sleeve Logo Jersey, L	49.99	L	Large
12	LJ-0192-X	Long-Sleeve Logo Jersey, XL	49.99	XL	Extra Large

Results | Messages

	ProductNumber	Name	ProductCategoryID	Category
1	BK-R93R-62	Road-150 Red, 62	6	Road Bikes
2	BK-R93R-44	Road-150 Red, 44	6	Road Bikes
3	BK-R93R-48	Road-150 Red, 48	6	Road Bikes
4	BK-R93R-52	Road-150 Red, 52	6	Road Bikes
5	BK-R93R-56	Road-150 Red, 56	6	Road Bikes
...				
23	BK-M82S-38	Mountain-100 Silver, 38	5	Mountain Bikes
24	BK-M82S-42	Mountain-100 Silver, 42	5	Mountain Bikes
25	BK-M82S-44	Mountain-100 Silver, 44	5	Mountain Bikes
...				
59	BK-T18U-54	Touring-3000 Blue, 54	7	Touring Bikes
60	BK-T18U-58	Touring-3000 Blue, 58	7	Touring Bikes
61	BK-T18U-62	Touring-3000 Blue, 62	7	Touring Bikes
62	BK-T18Y-44	Touring-3000 Yellow, ...	7	Touring Bikes
...				

Supporting Resources: **Querying Tables with SELECT**

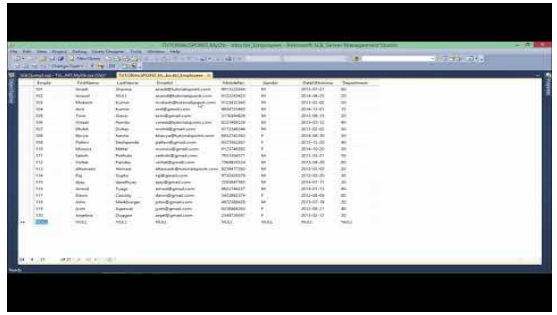
- **Select Statement Video link:**
  - YouTube | Select statement in sql server - Part 10

In this video we will learn

1. Select specific or all columns
2. Distinct rows
3. Filtering with where clause.
4. Wild Cards in SQL Server
5. Joining multiple conditions using AND and OR operators
6. Sorting rows using order by
7. Selecting top n or top n percentage of rows

T-SQL Variables in a SELECT QUERY

T-SQL - Select Statement



Supporting Resources: **Querying Tables with SELECT**

- **Select Statement Web Resource:**
  - Microsoft Docs | Select (Transact-SQL)
  - W3Schools | SQL Select Statement
  - SQL Server Tutorial.net | Basic SQL Server SELECT statement
  - Tutorialspoint | SQL - SELECT Database
  - JavaTPoint | T-SQL SELECT STATEMENT