

SQL Workshor (Dec):

Introduction to SQL –

Day 2

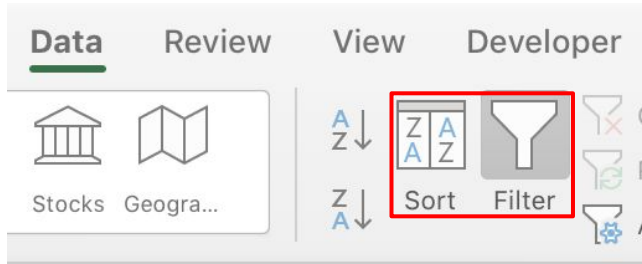
RECAP

- SQL data types
- Standard Query
- *
- Unique values
- Comparison operators
- (1,2,4,5)
- (1,2,3,4,5,6,7,8,9,10)
- Two fruits

SQL Syntax

- Not case sensitive
- Good practice:
 - capitalized the syntax and use small letters for the variables
 - Increase readability, makes debugging easier
 - Add semicolon “;” at the end of the query
 - Some SQL cannot work without “;” to tell it that the end of the query has been reached

EXCEL



Name	Number
A	3
B	
C	5
D	NA
E	NAN
F	#N/A
G	N/A
H	N.A

Sort

☒ A-Z Ascending ☐ Z-A Descending

By colour: None

Filter

By colour: None

Choose One

Search

- ☒ (Select All)
- ☒ 3
- ☒ 5
- ☒ N.A
- ☒ N/A
- ☒ NA
- ☒ NAN
- ☒ #N/A

Clear Filter

Filter Menu:

- ✓ Choose One
- Equals
- Does Not Equal
- Begin with
- Does Not Begin with
- Ends with
- Does Not End with
- Contains
- Does Not Contain**

EXCEL - COUNTIF

=COUNTIF(A1:A5,"*apple*")

cells that contain "apple"

=COUNTIF(A1:A5,"???")

cells that contain any 3 characters

LIKE - Specify patterns

- Only valid for String data types
- “%” represent zero, one or multiple characters
- “_” represent a single character
- Condition case sensitive (“e” is different from “E”)

Examples	
'T%'	Return values that starts with “T”
'%8'	Return values that ends with “8”
'_e%'	Return values that have “e” as the second character
'H____%'	Return values that starts with “H” and have at least 4 characters

LIKE - Specify patterns

SELECT * FROM production.products **WHERE** product_name **LIKE** 'T%'

	 product_id [PK] integer	 product_name character varying (255)	 brand_id integer	 category_id integer	 model_year smallint	 list_price numeric (10,2)
1	1	Trek 820 - 2016	9	6	2016	379.99
2	4	Trek Fuel EX 8 29 - 2016	9	6	2016	2899.99
3	7	Trek Slash 8 27.5 - 2016	9	6	2016	3999.99
4	8	Trek Remedy 29 Carbon Fram...	9	6	2016	1799.99
5	9	Trek Conduit+ - 2016	9	5	2016	2999.99
6	29	Trek X-Caliber 8 - 2017	9	6	2017	999.99
7	32	Trek Farley Alloy Frameset - 2...	9	6	2017	469.99
8	34	Trek Session DH 27.5 Carbon ...	9	6	2017	469.99
9	39	Trek Stache 5 - 2017	9	6	2017	1499.99







LIKE - Specify patterns

SELECT * **FROM** production.products **WHERE** product_name **LIKE** '%8'

	product_id [PK] integer	product_name character varying (255)	brand_id integer	category_id integer	model_year smallint	list_price numeric (10,2)
1	112	Trek 820 - 2018	9	6	2018	379.99
2	113	Trek Marlin 5 - 2018	9	6	2018	489.99
3	114	Trek Marlin 6 - 2018	9	6	2018	579.99
4	115	Trek Fuel EX 8 29 - 2018	9	6	2018	3199.99
5	116	Trek Marlin 7 - 2017/2018	9	6	2018	749.99
6	117	Trek Ticket S Frame - 2018	9	6	2018	1469.99
7	118	Trek X-Caliber 8 - 2018	9	6	2018	999.99
8	119	Trek Kids' Neko - 2018	9	6	2018	469.99
9	120	Trek Fuel EX 7 29 - 2018	9	6	2018	2499.99




LIKE - Specify patterns

SELECT * **FROM** production.products **WHERE** product_name **LIKE** '_e%'

	 product_id [PK] integer	 product_name character varying (255)	 brand_id integer	 category_id integer	 model_year smallint	 list_price numeric (10,2)
1	5	Heller Shagamaw Frame - 2016	3	6	2016	1320.99
2	131	Heller Bloodhound Trail - 2018	3	6	2018	2599.00
3	137	Heller Shagamaw GX1 - 2018	3	6	2018	2599.00

LIKE - Specify patterns

SELECT * **FROM** production.brands **WHERE** brand_name **LIKE** 'H____%'

	 brand_id [PK] integer 	brand_name  character varying (255)
1	2	Haro
2	3	Heller

HANDS-ON (LIKE)

1. Specify the LIKE pattern to find values that starts with 'S' and ends with '6'
2. Using an appropriate SQL statement, find the product_name of the products that contains 'nes'
3. Using an appropriate SQL statement, find the first_name and last_name of the customers who uses "aol" email address.
4. Using an appropriate SQL statement, find the staff_id of the staffs whose last_name is 5 character long.

IS NULL / NOT - Empty values / Opposites

SELECT * FROM production.brands **WHERE** brand_name **IS NULL**

SELECT * FROM production.brands **WHERE** brand_name **IS NOT NULL**

	brand_id [PK] integer	brand_name character varying (255)
1	1	Electra
2	2	Haro
3	3	Heller
4	4	Pure Cycles
5	5	Ritchey
6	6	Strider
7	7	Sun Bicycles
8	8	Surly
9	9	Trek

	brand_id [PK] integer	brand_name character varying (255)

If returned value is empty, it means that there is no *NULL* value in "brand_name"

ORDER BY

- Sorts the returned values in ascending or descending order
- Default sort type is ascending

ORDER BY

SELECT * FROM production.stocks **WHERE** (store_id = 1 **OR** store_id = 2) **AND** quantity = 0 **ORDER BY** product_id

	store_id [PK] integer	product_id [PK] integer	quantity integer
1	1	6	0
2	1	8	0
3	2	22	0
4	1	32	0
5	1	42	0
6	2	47	0
7	2	91	0
8	1	92	0
9	2	158	0
10	1	160	0
11	1	163	0
12	1	168	0
13	2	175	0
14	2	184	0

SELECT * FROM production.stocks **WHERE** (store_id = 1 **OR** store_id = 2) **AND** quantity = 0 **ORDER BY** product_id **DESC**

	store_id [PK] integer	product_id [PK] integer	quantity integer
1	1	302	0
2	2	299	0
3	2	251	0
4	1	246	0
5	2	198	0
6	2	192	0
7	2	184	0
8	2	175	0
9	1	168	0
10	1	163	0
11	1	160	0
12	2	158	0
13	1	92	0
14	2	91	0

LIMIT

- **ALWAYS** last part of the query
- Specify the number of rows to display

SELECT * FROM production.brands 

SELECT * FROM production.brands **LIMIT 5** 

	brand_id [PK] integer	brand_name character varying (255)
1	1	Electra
2	2	Haro
3	3	Heller
4	4	Pure Cycles
5	5	Ritchey

	brand_id [PK] integer	brand_name character varying (255)
1	1	Electra
2	2	Haro
3	3	Heller
4	4	Pure Cycles
5	5	Ritchey
6	6	Strider
7	7	Sun Bicycles
8	8	Surly
9	9	Trek

LIMIT

SELECT * FROM production.stocks **WHERE** (store_id = 1 **OR** store_id = 2)
AND quantity = 0 **ORDER BY** product_id **DESC** **LIMIT** 10

	store_id [PK] integer	product_id [PK] integer	quantity integer
1	1	302	0
2	2	299	0
3	2	251	0
4	1	246	0
5	2	198	0
6	2	192	0
7	2	184	0
8	2	175	0
9	1	168	0
10	1	163	0
11	1	160	0
12	2	158	0
13	1	92	0
14	2	91	0



	store_id [PK] integer	product_id [PK] integer	quantity integer
1	1	302	0
2	2	299	0
3	2	251	0
4	1	246	0
5	2	198	0
6	2	192	0
7	2	184	0
8	2	175	0
9	1	168	0
10	1	163	0

HANDS-ON (IS NULL/NOT, ORDER BY, LIMIT)

1. Using an appropriate SQL statement, find the email of those customers that did not indicate their phone number.
2. Using an appropriate SQL statement, find the shipped_date of the orders that had been shipped out.
3. Using an appropriate SQL statement, sort the customers table with city in descending order and first_name in ascending order.
4. Using an appropriate SQL statement, find the top 3 most expensive product whose model_year = 2019.

END OF DAY 2!

Any questions? Feel free to clarify now.

Or you can reach us at:

Wei Teck (Jensen): wtlow003@suss.edu.sg /
<https://www.linkedin.com/in/weitecklow/>

Jeanette: jeanettekoh001@suss.edu.sg /
[linkedin.com/in/jeanette-koh-872b64192](https://www.linkedin.com/in/jeanette-koh-872b64192)