Learn SQL by Example from Basic to Advanced

Chananel Perel 2023 © All Rights Reserved

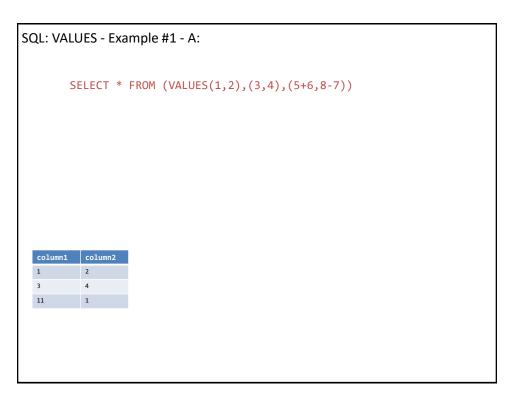
Learn SOL by Example (Chananel Perel) 2024-04-10 16:17:23 404279

SQL READ 3 CONT.



```
SQL: VALUES - Example #1 - Q:

SELECT * FROM (VALUES(1,2),(3,4),(5+6,8-7))
```





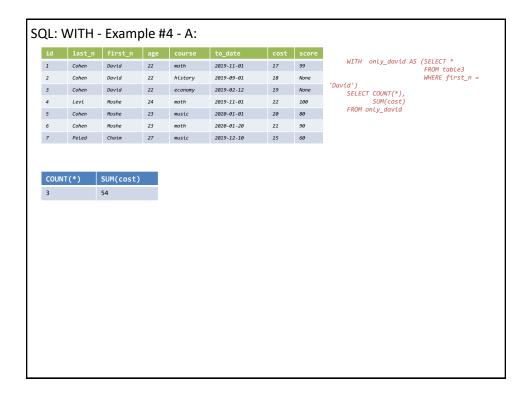
```
SQL: WITH tbl1(col1,col2) AS (VALUES(1,2),(3,4),(5+6,8-7))
SELECT *
FROM tbl1
```

```
SQL: WITH - Example #2 - A:

WITH tbl1(col1,col2) AS (VALUES(1,2),(3,4),(5+6,8-7))
SELECT *
FROM tbl1

Col1 col2
1 2
3 4
11 1
```

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	None
3	Cohen	David	22	economy	2019-02-12	19	None
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	music	2019-12-10	15	60
	SELECT	only_dav 「COUNT(* SUM(cos only_davi), t)	FROM t		avid')



GROUP BY

SQL: GROUP BY - Example #5 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

SQL: GROUP BY - Example #5 - A:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	PeLed	Chaim	27	history	2019-12-10	15	60
8	PeLed	Chaim	27	math	2019-10-20	16	70
q	Gami I	David	25	hictory	2010-00-20	17	oc

SELECT course, COUNT(*), AVG(score) FROM table1 GROUP BY course ORDER BY 1

course	COUNT(*)	AVG(score)
economy	1	77.0
history	3	77.6666666666667
math	4	89.75
music	1	80.0

SQL: GROUP BY - Example #6 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

```
SELECT course,

MAX(cost) max_cost,

MIN(age) min_age,

ROUND(AVG(score),2) avg_score,

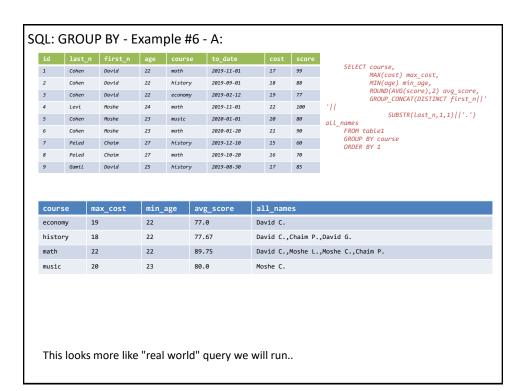
GROUP_CONCAT(DISTINCT first_n||' '||

SUBSTR(last_n,1,1)||'.') all_names

FROM table1

GROUP BY course

ORDER BY 1
```



SQL: 0	GROUP B	Y - Examp	le #7	7 - Q:			
id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85
	FROM 1	T first_n AVG(sco SUM(cos table1 BY first BY 1,2	re), t)				

SQL: GROUP BY - Example #7 - A:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	PeLed	Chaim	27	history	2019-12-10	15	60
8	PeLed	Chaim	27	math	2019-10-20	16	70
q	Gami I	David	25	hictory	2010-00-20	17	oc

SELECT first_n, last_n,
 AVG(score),
 SUM(cost)
FROM tabLe1
GROUP BY first_n, last_n
ORDER BY 1,2

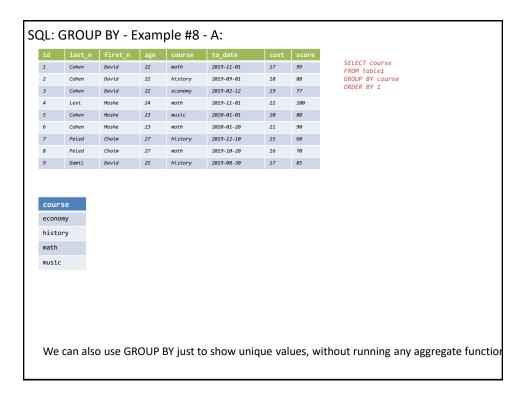
first_n	last_n	AVG(score)	SUM(cost)
Chaim	Peled	65.0	31
David	Cohen	88.0	54
David	Gamil	85.0	17
Moshe	Cohen	85.0	41
Moshe	Levi	100.0	22

We can GROUP BY more than one column

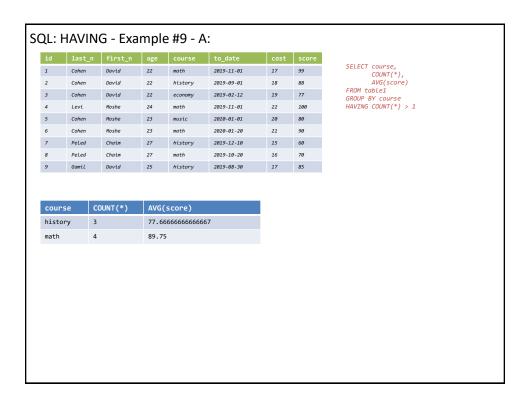
SQL: GROUP BY - Example #8 - Q:

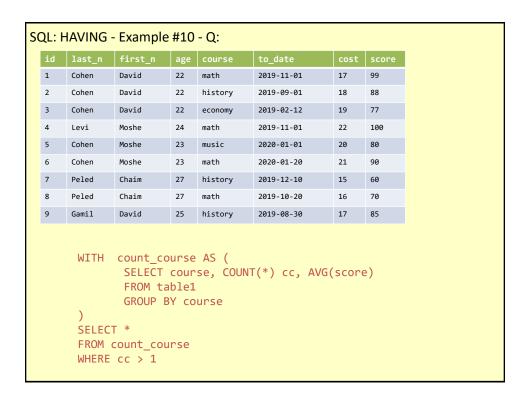
id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

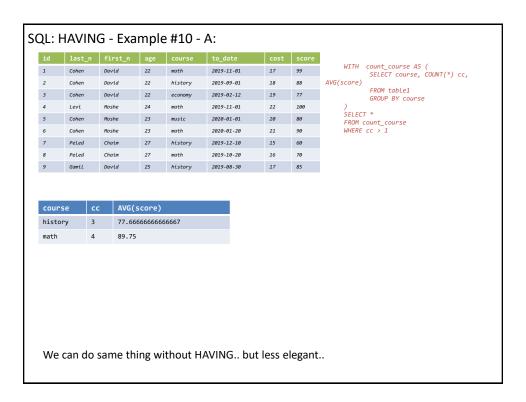
SELECT course FROM table1 GROUP BY course ORDER BY 1





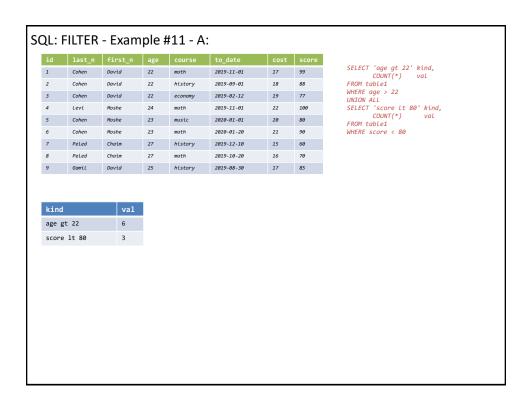




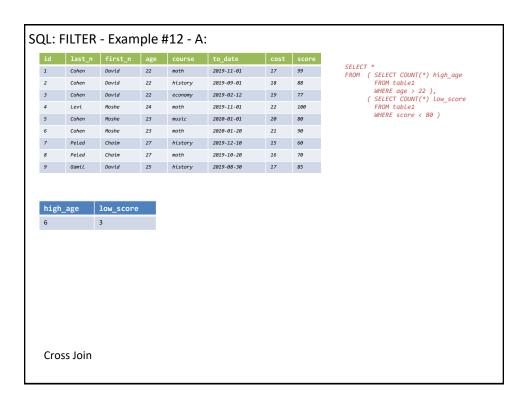


FILTER

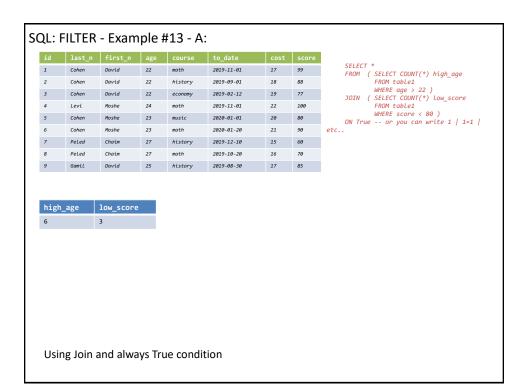
QL: F	ILTER - E	xample #	11 -	Q:			
id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85
	FROM 1 WHERE UNION SELECT	<pre>r 'age gt COUNT(* table1 age > 22 ALL r 'score COUNT(* table1 score <</pre>) lt 8)	val 0' kind,			



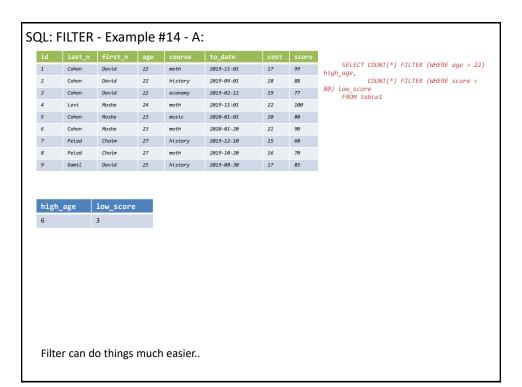
```
SQL: FILTER - Example #12 - Q:
  1
       Cohen
                 David
                            22
                                 math
                                             2019-11-01
                                                            17
                                                                   99
  2
        Cohen
                 David
                            22
                                 history
                                             2019-09-01
                                                                   88
       Cohen
                 David
                            22
                                             2019-02-12
  3
                                 economy
                                                            19
                                                                   77
  4
        Levi
                 Moshe
                            24
                                 math
                                             2019-11-01
                                                            22
                                                                   100
  5
        Cohen
                 Moshe
                            23
                                 music
                                             2020-01-01
                                                            20
                                                                   80
  6
        Cohen
                 Moshe
                            23
                                 math
                                             2020-01-20
                                                            21
                                                                   90
  7
        Peled
                 Chaim
                            27
                                 history
                                             2019-12-10
                                                            15
                                                                   60
                                                                   70
  8
        Peled
                 Chaim
                            27
                                 math
                                             2019-10-20
                                                            16
        Gamil
  9
                 David
                            25
                                 history
                                             2019-08-30
         SELECT *
         FROM ( SELECT COUNT(*) high_age
                   FROM table1
                   WHERE age > 22 ),
                 ( SELECT COUNT(*) low_score
                   FROM table1
                   WHERE score < 80 )
```

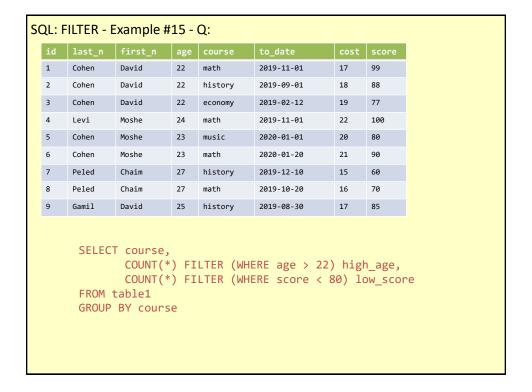


```
SQL: FILTER - Example #13 - Q:
  1
       Cohen
                 David
                           22
                                           2019-11-01
                                                           17
                               math
                                                                 99
  2
       Cohen
                 David
                           22
                                history
                                           2019-09-01
                                                                 88
       Cohen
                David
                           22
                                           2019-02-12
  3
                                economy
                                                           19
                                                                 77
  4
       Levi
                 Moshe
                           24
                                math
                                           2019-11-01
                                                           22
                                                                 100
  5
       Cohen
                 Moshe
                           23
                                music
                                           2020-01-01
                                                           20
                                                                 80
  6
       Cohen
                 Moshe
                           23
                                math
                                           2020-01-20
                                                           21
                                                                 90
  7
       Peled
                 Chaim
                           27
                                history
                                           2019-12-10
                                                           15
                                                                 60
  8
       Peled
                 Chaim
                           27
                                math
                                           2019-10-20
                                                           16
                                                                 70
  9
       Gamil
                 David
                           25
                                history
                                           2019-08-30
         SELECT *
         FROM ( SELECT COUNT(*) high_age
                   FROM table1
                   WHERE age > 22 )
         JOIN ( SELECT COUNT(*) low_score
                   FROM table1
                   WHERE score < 80 )
         ON True -- or you can write 1 | 1=1 | etc..
```



SQL: FILTER - Example #14 - Q: 1 Cohen David 22 2019-11-01 17 math 99 2 Cohen David 22 history 2019-09-01 88 Cohen David 22 2019-02-12 3 economy 19 77 4 Levi Moshe 24 math 2019-11-01 22 100 5 Cohen Moshe 23 music 2020-01-01 20 80 6 Cohen Moshe 23 math 2020-01-20 21 90 7 Peled Chaim 27 2019-12-10 history 15 60 2019-10-20 8 Peled Chaim 27 math 16 70 9 Gamil David 25 history 2019-08-30 SELECT COUNT(*) FILTER (WHERE age > 22) high_age, COUNT(*) FILTER (WHERE score < 80) low score FROM table1





		first_n	age		to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	PeLed	Chaim	27	history	2019-12-10	15	60
8	PeLed	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

SELECT course,
COUNT(*) FILTER (WHERE age > 22)
high_age,
COUNT(*) FILTER (WHERE score <
80) Low_score
FROM table1
GROUP BY course

course	high_age	low_score
economy	0	1
history	2	1
math	3	1
music	1	0

We can also do this within groups..

SQL: FILTER - Example #16 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

```
SELECT course,

COUNT(CASE WHEN age > 22 THEN 1 ELSE NULL END)

high_age,

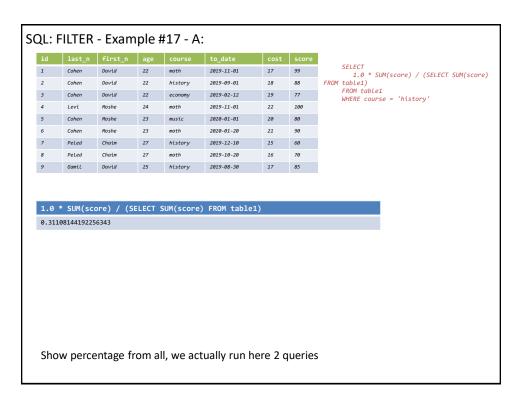
COUNT(CASE WHEN score < 80 THEN 1 END) low_score

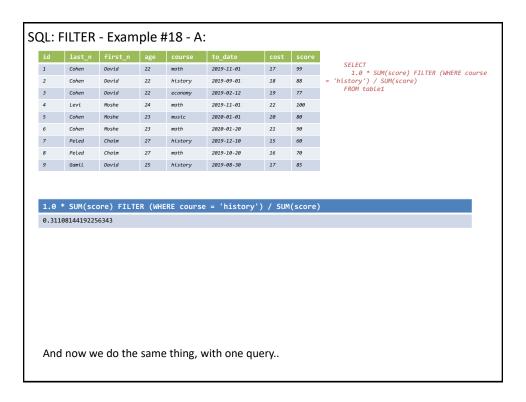
FROM table1

GROUP BY course
```

SQL: FILTER - Example #16 - A: id last_n first_n age course to_date SELECT course, COUNT(CASE WHEN age > 22 THEN 1 1 Cohen David 22 math 2019-11-01 17 99 ELSE NULL END) high_age, COUNT(CASE WHEN score < 80 THEN 1 2 Cohen David 22 history 2019-09-01 18 88 3 Cohen David 22 economy 2019-02-12 19 77 END) low_score FROM table1 Moshe 24 math 22 100 Levi 2019-11-01 GROUP BY course 23 music 2020-01-01 20 80 5 Cohen Moshe 23 math 21 90 PeLed Chaim 27 history 15 60 2019-12-10 8 PeLed Chaim 27 math 2019-10-20 16 70 9 Gamil David 25 history 2019-08-30 17 85 high_age low_score 1 economy math music In some version FILTER is not supported, so we can use CASE WHEN

SQL: FILTER - Example #17 - Q: 1 Cohen David 22 math 2019-11-01 17 99 2 Cohen David 22 history 2019-09-01 88 22 economy Cohen David 2019-02-12 3 19 77 4 Levi Moshe 24 math 2019-11-01 22 100 5 Cohen Moshe 23 music 2020-01-01 20 80 6 Cohen Moshe 23 math 2020-01-20 21 90 7 Peled Chaim 27 history 2019-12-10 15 60 70 8 Peled Chaim 27 math 2019-10-20 16 Gamil 9 David 25 history 2019-08-30 17 **SELECT** 1.0 * SUM(score) / (SELECT SUM(score) FROM table1) FROM table1 WHERE course = 'history'





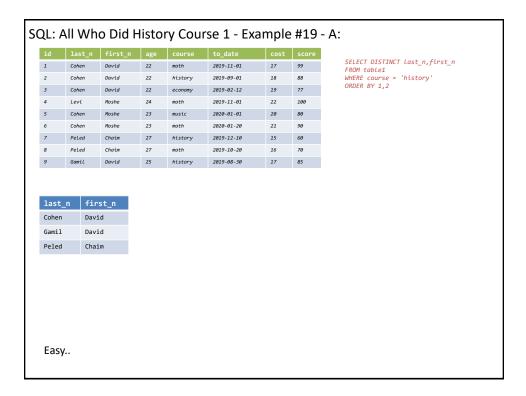


ALL WHO DID HISTORY COURSE 1

SQL: All Who Did History Course 1 - Example #19 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

SELECT DISTINCT last_n,first_n
FROM table1
WHERE course = 'history'
ORDER BY 1,2

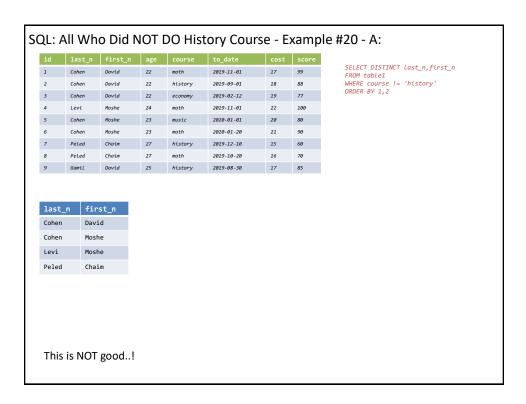


Learn SQL by Example (Chananel Perel 2023)

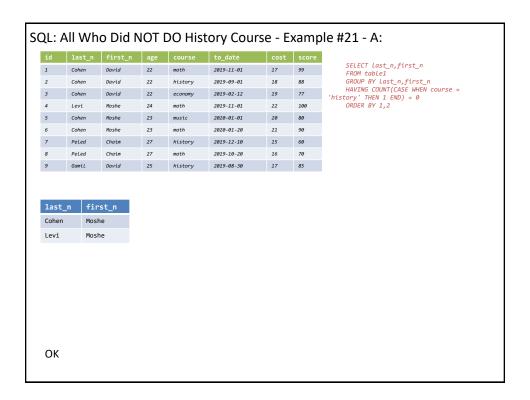
ALL WHO DID NOT DO HISTORY

COURSE

	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85
	FROM 1	T DISTINC table1 course ! BY 1,2			st_n		



Ld	last_n	first_n	age	course	to_date	cost	score	
L	Cohen	David	22	math	2019-11-01	17	99	
2	Cohen	David	22	history	2019-09-01	18	88	
;	Cohen	David	22	economy	2019-02-12	19	77	
	Levi	Moshe	24	math	2019-11-01	22	100	
;	Cohen	Moshe	23	music	2020-01-01	20	80	
5	Cohen	Moshe	23	math	2020-01-20	21	90	
•	Peled	Chaim	27	history	2019-12-10	15	60	
;	Peled	Chaim	27	math	2019-10-20	16	70	
)	Gamil	David	25	history	2019-08-30	17	85	
	CELEC:	Γ last n,	firs	t_n				

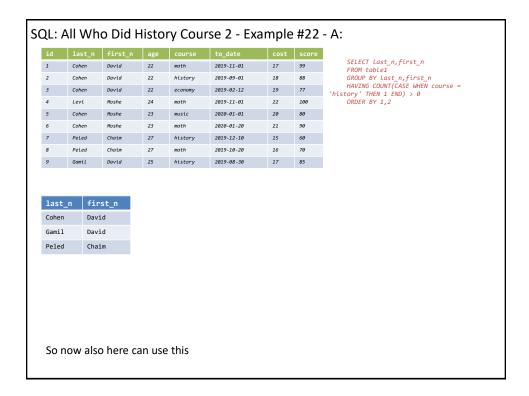


ALL WHO DID HISTORY COURSE 2

SQL: All Who Did History Course 2 - Example #22 - Q:

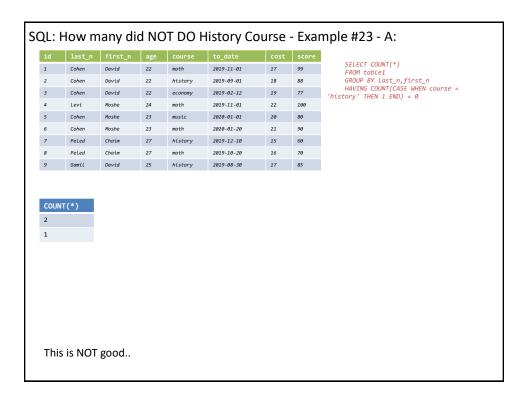
id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

SELECT last_n,first_n
FROM table1
GROUP BY last_n,first_n
HAVING COUNT(CASE WHEN course = 'history' THEN 1 END) > 0
ORDER BY 1,2

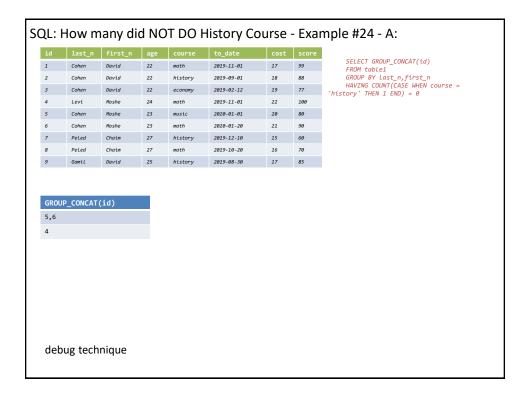


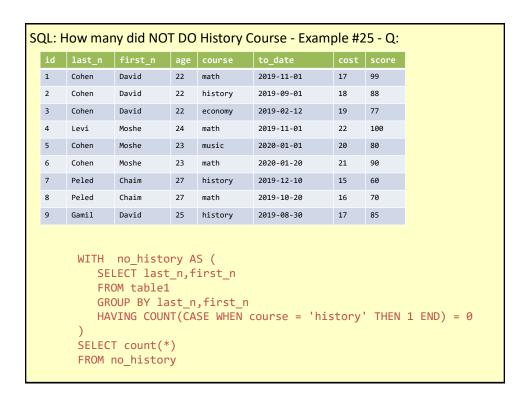
Learn SQL by Example (Chananel Perel 2023)
HOW MANY DID NOT DO HISTORY
COURSE

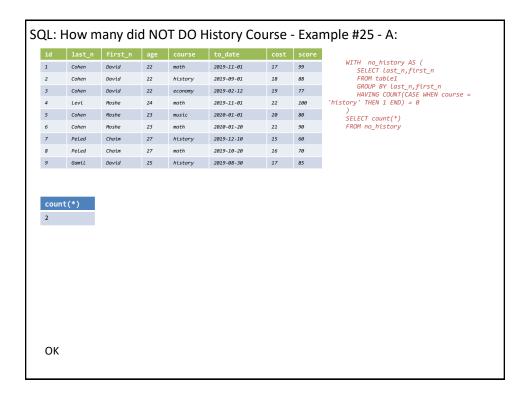
l last_ Cohen	n first_n David	age 22	course	to_date 2019-11-01	cost	score	
Cohen	David	22	history	2019-09-01	18	88	
Cohen	David	22	economy	2019-02-12	19	77	
Levi	Moshe	24	math	2019-11-01	22	100	
Cohen	Moshe	23	music	2020-01-01	20	80	
Cohen	Moshe	23	math	2020-01-20	21	90	
Peled	Chaim	27	history	2019-12-10	15	60	
Peled	Chaim	27	math	2019-10-20	16	70	
Gamil	David	25	history	2019-08-30	17	85	
	ECT COUNT(M table1	*)					



last_n	first_n	age	course	to_date	cost	score
Cohen	David	22	math	2019-11-01	17	99
Cohen	David	22	history	2019-09-01	18	88
Cohen	David	22	economy	2019-02-12	19	77
Levi	Moshe	24	math	2019-11-01	22	100
Cohen	Moshe	23	music	2020-01-01	20	80
Cohen	Moshe	23	math	2020-01-20	21	90
Peled	Chaim	27	history	2019-12-10	15	60
Peled	Chaim	27	math	2019-10-20	16	70
Gamil	David	25	history	2019-08-30	17	85
FROM t	BY last_	n,fi	rst_n	rse = 'histor	y' T⊦	IEN 1 E





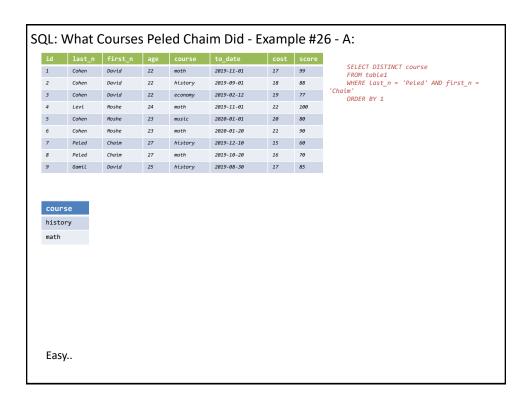


WHAT COURSES PELED CHAIM DID

SQL: What Courses Peled Chaim Did - Example #26 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

```
SELECT DISTINCT course
FROM table1
WHERE last_n = 'Peled' AND first_n = 'Chaim'
ORDER BY 1
```

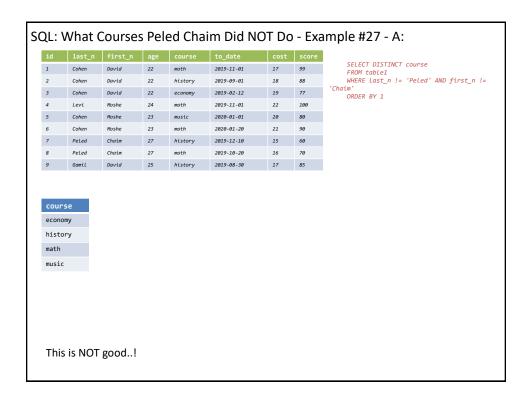


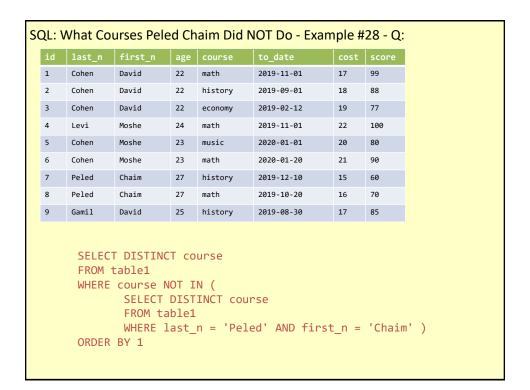
Learn SQL by Example (Chananel Perel 2023)

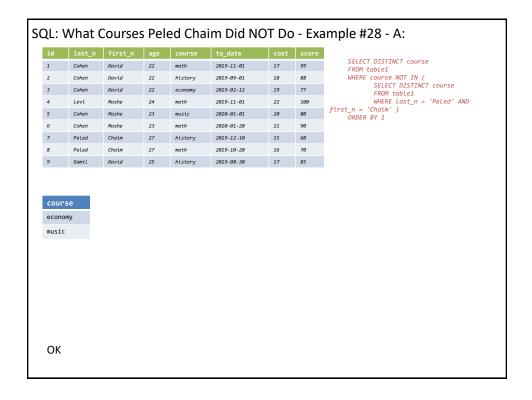
WHAT COURSES PELED CHAIM DID

NOT DO

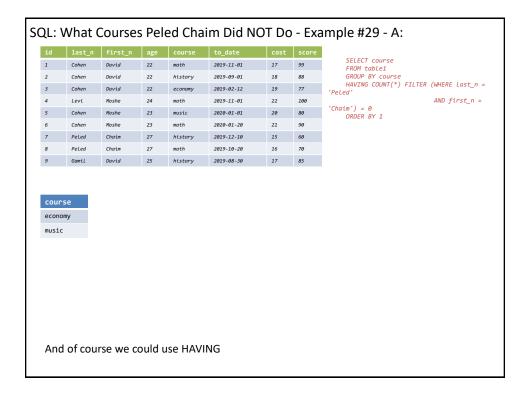
l last_n	first_n	age	course	to_date	cost	score
L Cohen	David	22	math	2019-11-01	17	99
Cohen	David	22	history	2019-09-01	18	88
3 Cohen	David	22	economy	2019-02-12	19	77
l Levi	Moshe	24	math	2019-11-01	22	100
Cohen	Moshe	23	music	2020-01-01	20	80
Cohen	Moshe	23	math	2020-01-20	21	90
7 Peled	Chaim	27	history	2019-12-10	15	60
B Peled	Chaim	27	math	2019-10-20	16	70
Gamil	David	25	history	2019-08-30	17	85
FROM	_) first_n !=	'Chai	im'







	last_n	first_n	age	course	to_date	cost	score		
1	Cohen	David	22	math	2019-11-01	17	99		
2	Cohen	David	22	history	2019-09-01	18	88		
3	Cohen	David	22	economy	2019-02-12	19	77		
4	Levi	Moshe	24	math	2019-11-01	22	100		
5	Cohen	Moshe	23	music	2020-01-01	20	80		
6	Cohen	Moshe	23	math	2020-01-20	21	90		
7	Peled	Chaim	27	history	2019-12-10	15	60		
8	Peled	Chaim	27	math	2019-10-20	16	70		
9	Gamil	David	25	history	2019-08-30	17	85		

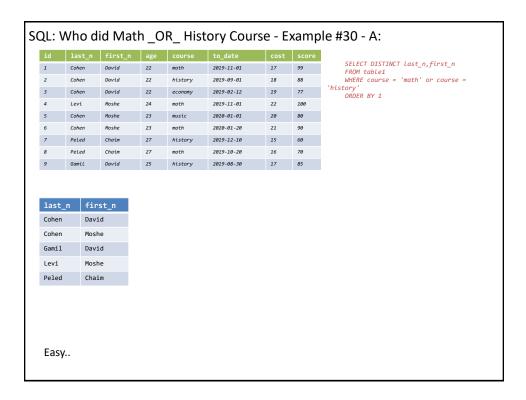


WHO DID MATH _OR_ HISTORY COURSE

SQL: Who did Math OR History Course - Example #30 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

```
SELECT DISTINCT last_n,first_n
FROM table1
WHERE course = 'math' or course = 'history'
ORDER BY 1
```



Learn SQL by Example (Chananel Perel 2023)

WHO DID MATH _AND_ HISTORY

COURSE

SQL: Who did Math	_AND_	History Course -	Example #31 - Q:
-------------------	-------	------------------	------------------

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

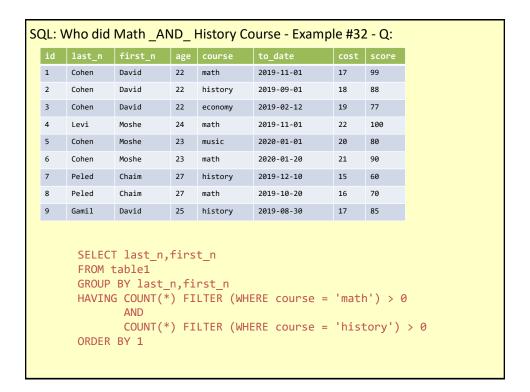
```
SELECT DISTINCT last_n,first_n
FROM table1
WHERE course = 'math' and course = 'history'
ORDER BY 1
```

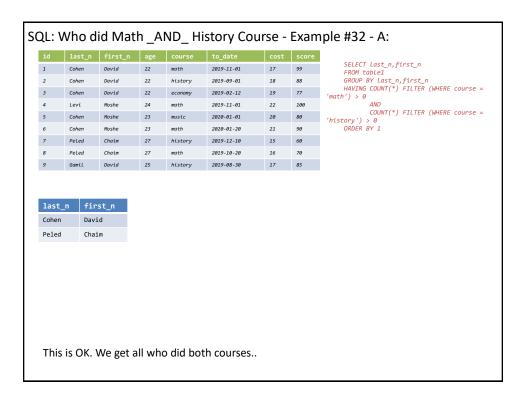
SQL: Who did Math _AND_ History Course - Example #31 - A:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	PeLed	Chaim	27	history	2019-12-10	15	60
8	PeLed	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

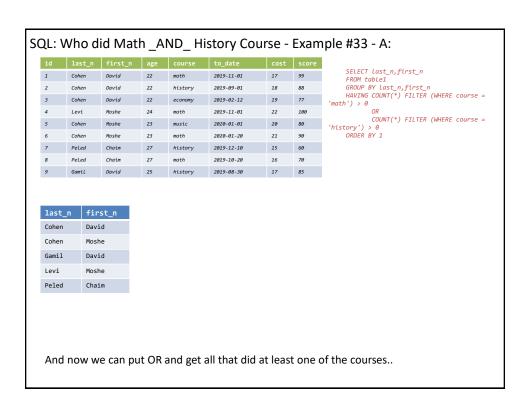
SELECT DISTINCT last_n,first_n FROM table1 WHERE course = 'math' and course = 'history' ORDER BY 1

We get nothing (always).. This is NOT good..!





SQL: Who did Math _AND_ History Course - Example #33 - Q: age course to date 2019-11-01 1 Cohen David math 17 99 David 22 history 2019-09-01 18 88 2 Cohen 3 Cohen David 22 economy 2019-02-12 19 77 2019-11-01 4 Levi Moshe 24 math 22 100 5 Cohen Moshe 23 music 2020-01-01 20 80 6 Cohen Moshe 23 math 2020-01-20 21 90 7 Peled Chaim 27 history 2019-12-10 15 60 8 Peled 27 math 2019-10-20 70 9 David Gamil 25 history 2019-08-30 17 85 SELECT last_n,first_n FROM table1 GROUP BY last n, first n HAVING COUNT(*) FILTER (WHERE course = 'math') > 0 COUNT(*) FILTER (WHERE course = 'history') > 0 ORDER BY 1

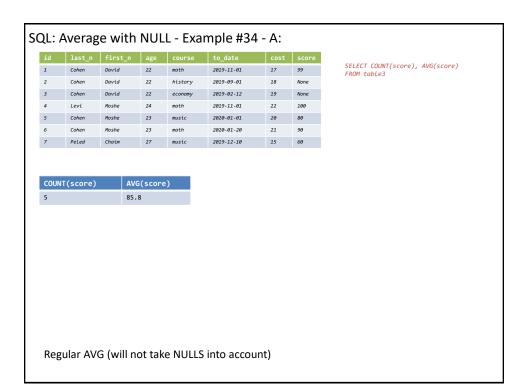


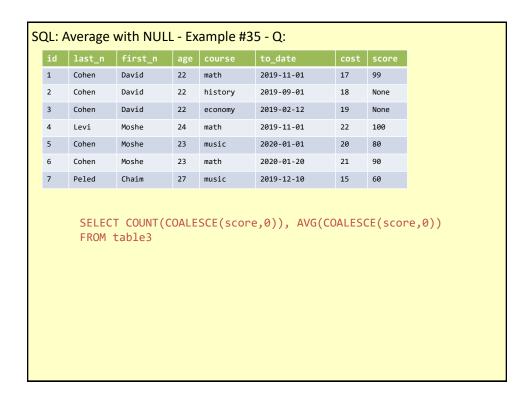
AVERAGE WITH NULL

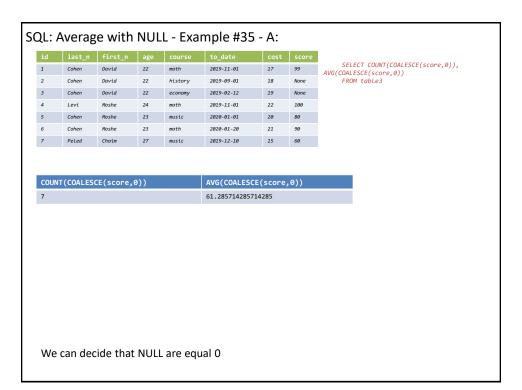
SQL: Average with NULL - Example #34 - Q:

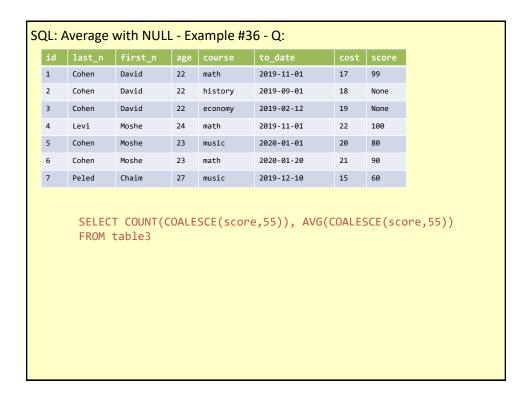
id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	None
3	Cohen	David	22	economy	2019-02-12	19	None
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	music	2019-12-10	15	60

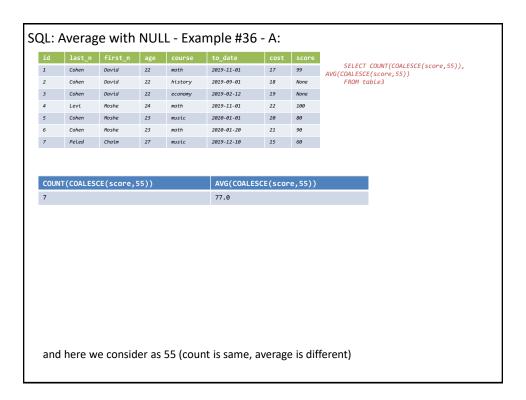
SELECT COUNT(score), AVG(score)
FROM table3

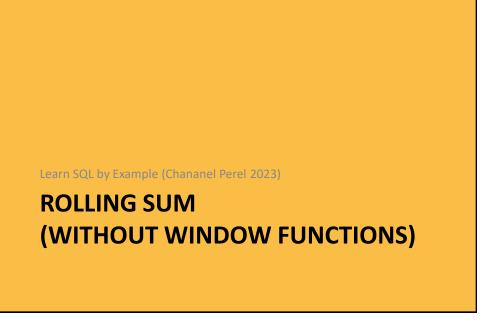




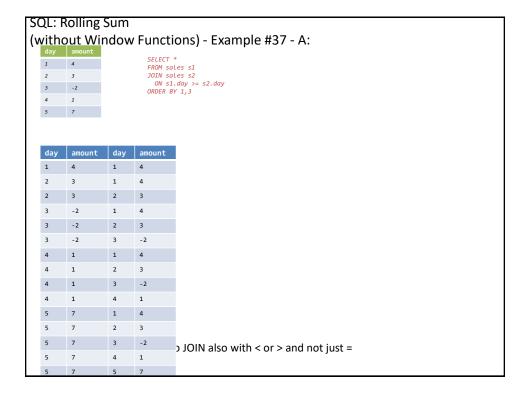


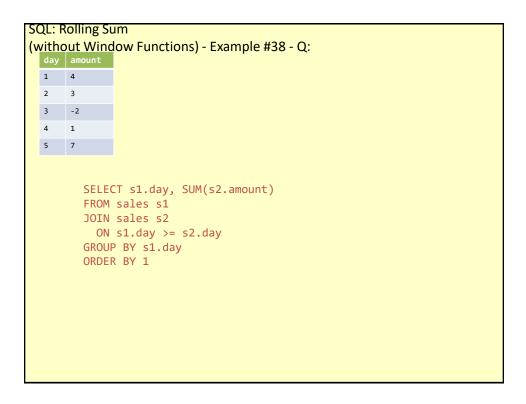


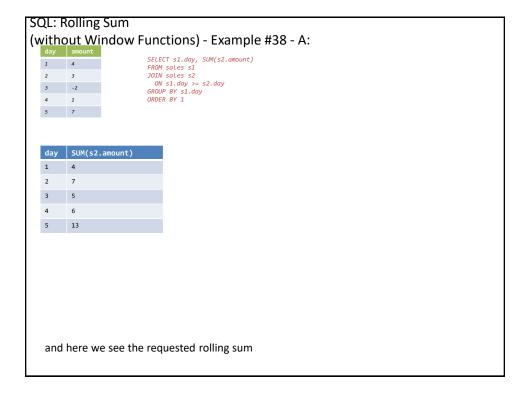


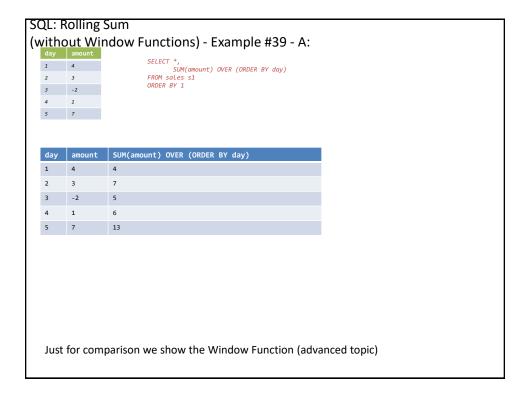


```
SQL: Rolling Sum
(without Window Functions) - Example #37 - Q:
  1
      4
  2
      3
  3
      -2
  4
      1
  5
      7
        SELECT *
        FROM sales s1
        JOIN sales s2
          ON s1.day >= s2.day
        ORDER BY 1,3
```







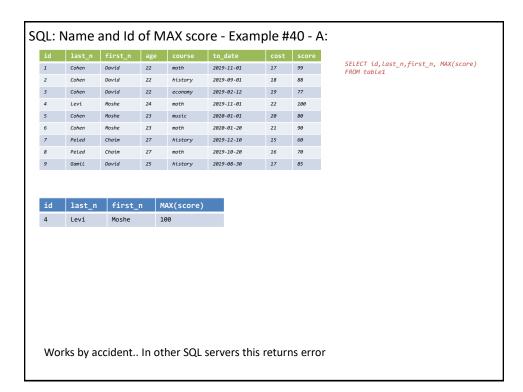


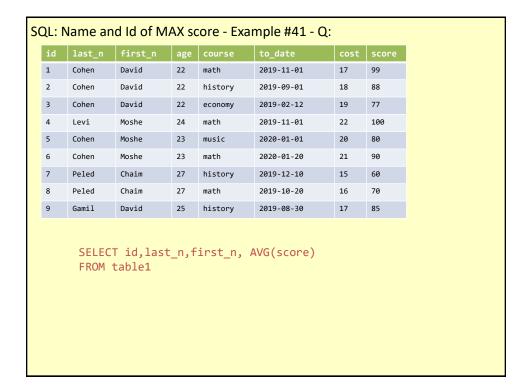
NAME AND ID OF MAX SCORE

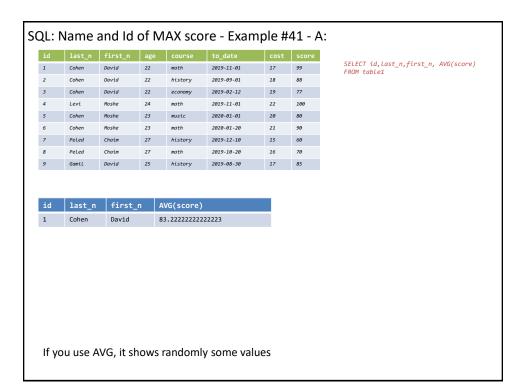
SQL: Name and Id of MAX score - Example #40 - Q:

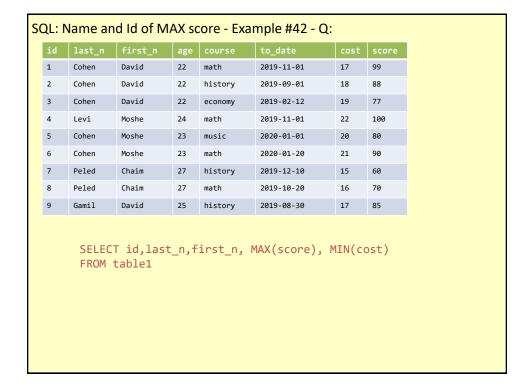
id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

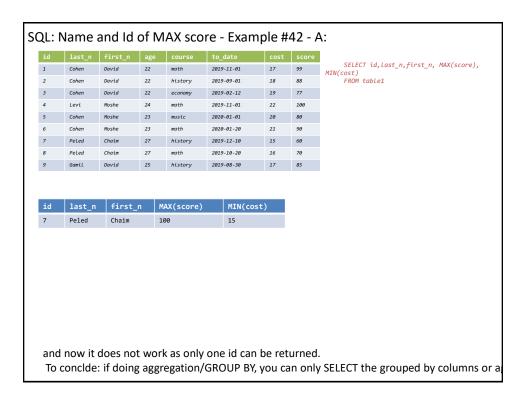
SELECT id,last_n,first_n, MAX(score)
FROM table1

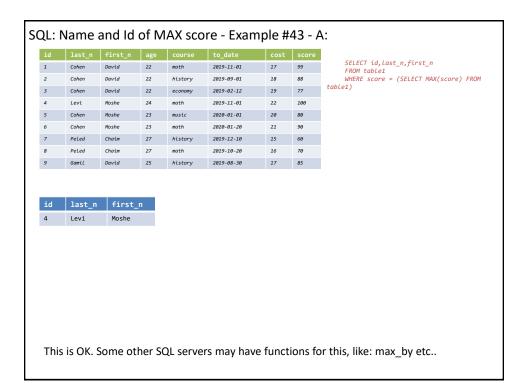


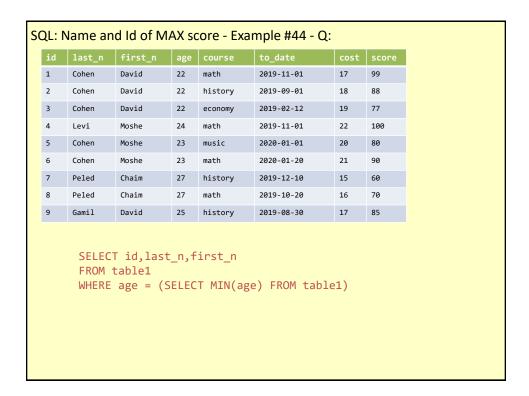


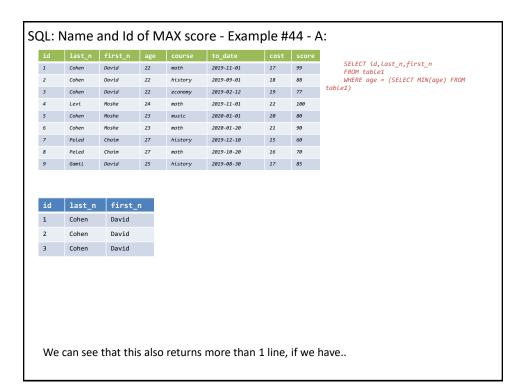












Learn SQL by Example (Chananel Perel 2023)

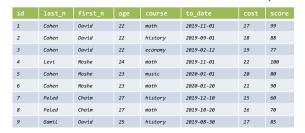
NAME AND ID OF 2 HIGHEST SCORE

SQL: Name and Id of 2 HIGHEST score - Example #45 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

SELECT id,last_n,first_n, score FROM table1 ORDER BY score DESC LIMIT 2

SQL: Name and Id of 2 HIGHEST score - Example #45 - A:



SELECT id,Last_n,first_n, score FROM table1 ORDER BY score DESC LIMIT 2

id	last_n	first_n	score
4	Levi	Moshe	100
1	Cohen	David	99

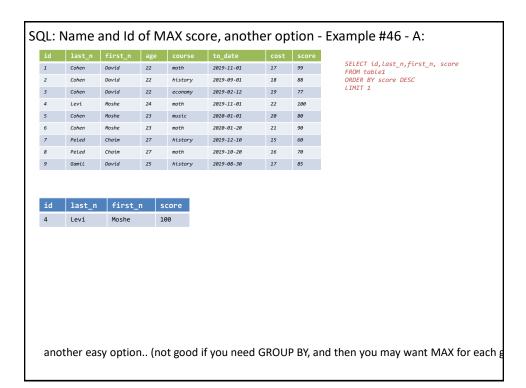
doing like this is even easier, but good only for MIN, MAX, and N highest/lowest.. (not good for average e.g.)

NAME AND ID OF MAX SCORE, ANOTHER OPTION

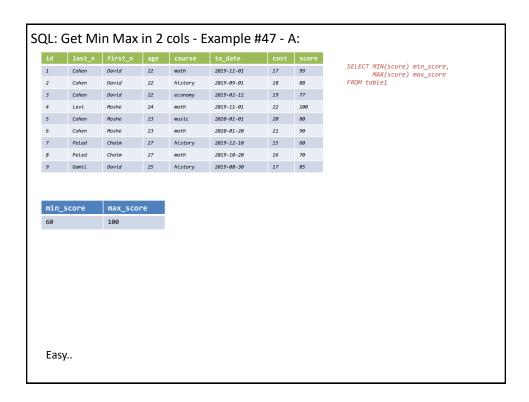
SQL: Name and Id of MAX score, another option - Example #46 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

SELECT id,last_n,first_n, score FROM table1 ORDER BY score DESC LIMIT 1





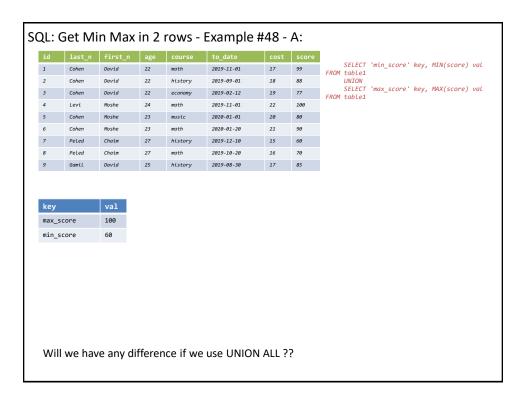


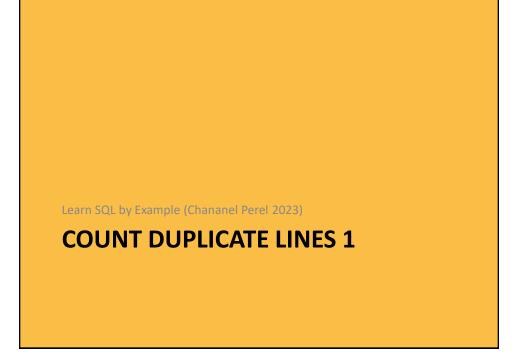
GET MIN MAX IN 2 ROWS

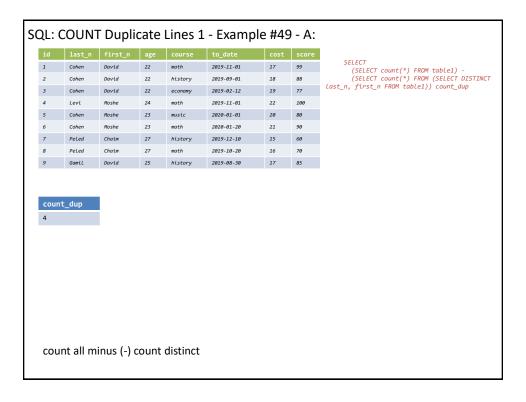
SQL: Get Min Max in 2 rows - Example #48 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

SELECT 'min_score' key, MIN(score) val FROM table1 UNION SELECT 'max_score' key, MAX(score) val FROM table1







COUNT DUPLICATE LINES 2

SQL: COUNT Duplicate Lines 2 - Example #50 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

SELECT last_n, first_n
FROM table1
GROUP BY last_n, first_n
HAVING COUNT(*) > 1

SQL: COUNT Duplicate Lines 2 - Example #50 - A:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	PeLed	Chaim	27	history	2019-12-10	15	60
8	PeLed	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

SELECT Last_n, first_n FROM table1 GROUP BY Last_n, first_n HAVING COUNT(*) > 1

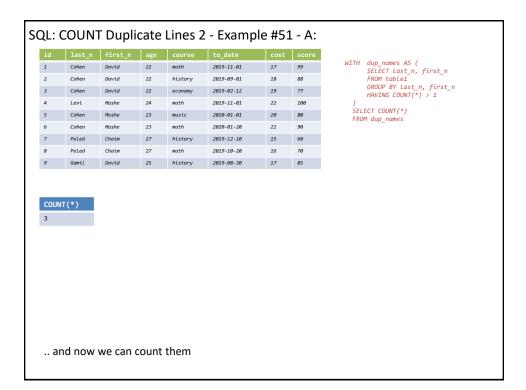
last_n	first_n
Cohen	David
Cohen	Moshe
Peled	Chaim

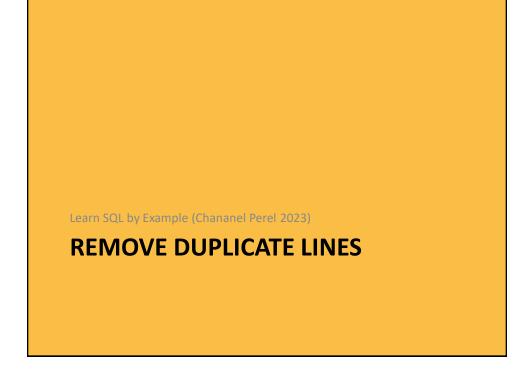
Here we can see the duplicate names we have

SQL: COUNT Duplicate Lines 2 - Example #51 - Q:

id	last_n	first_n	age	course	to_date	cost	score
1	Cohen	David	22	math	2019-11-01	17	99
2	Cohen	David	22	history	2019-09-01	18	88
3	Cohen	David	22	economy	2019-02-12	19	77
4	Levi	Moshe	24	math	2019-11-01	22	100
5	Cohen	Moshe	23	music	2020-01-01	20	80
6	Cohen	Moshe	23	math	2020-01-20	21	90
7	Peled	Chaim	27	history	2019-12-10	15	60
8	Peled	Chaim	27	math	2019-10-20	16	70
9	Gamil	David	25	history	2019-08-30	17	85

```
WITH dup_names AS (
          SELECT last_n, first_n
          FROM table1
          GROUP BY last_n, first_n
          HAVING COUNT(*) > 1
)
SELECT COUNT(*)
FROM dup_names
```

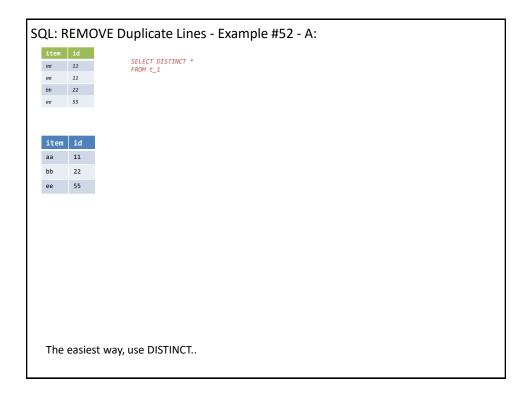




```
SQL: REMOVE Duplicate Lines - Example #52 - Q:

item id
aa 11
bb 22
ee 55

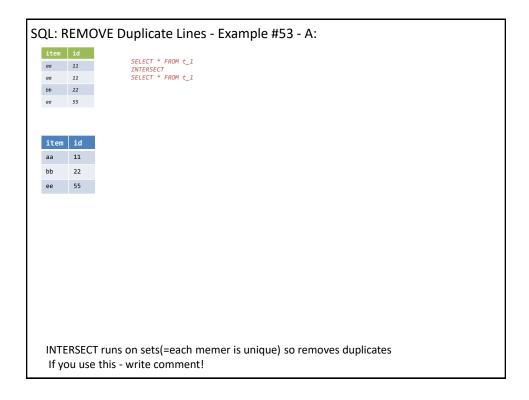
SELECT DISTINCT *
FROM t_1
```



```
SQL: REMOVE Duplicate Lines - Example #53 - Q:

item id
aa 11
bb 22
ee 55

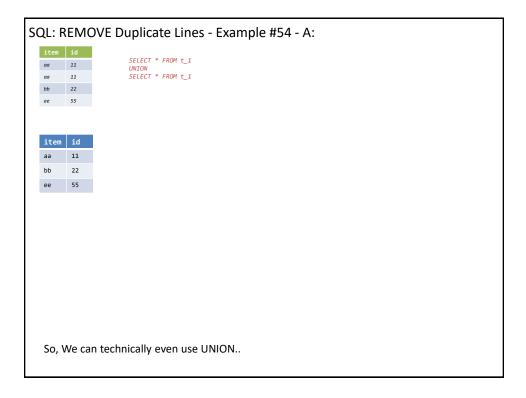
SELECT * FROM t_1
INTERSECT
SELECT * FROM t_1
```



```
SQL: REMOVE Duplicate Lines - Example #54 - Q:

item id
aa 11
bb 22
ee 55

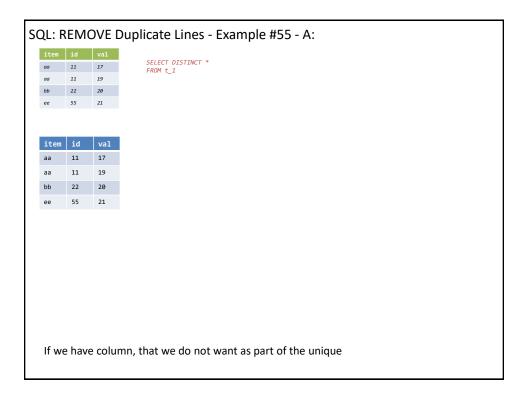
SELECT * FROM t_1
UNION
SELECT * FROM t_1
```

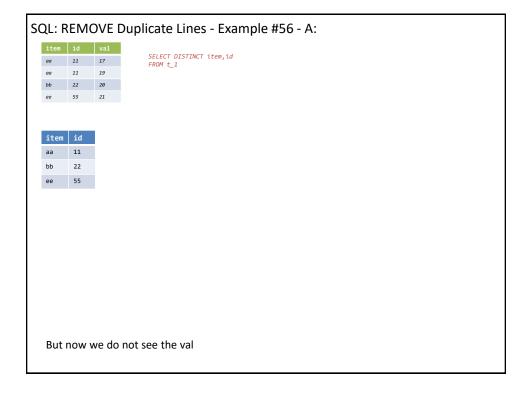


```
SQL: REMOVE Duplicate Lines - Example #55 - Q:

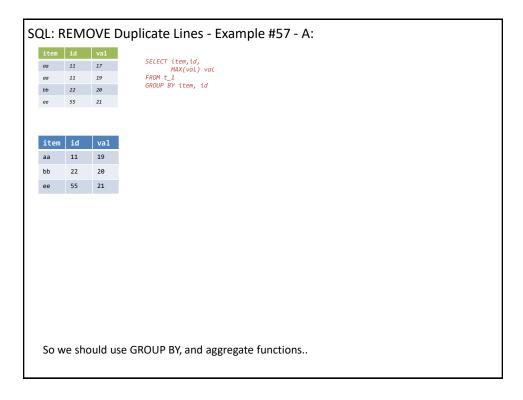
| item | id | val |
| aa | 11 | 17 |
| aa | 11 | 19 |
| bb | 22 | 28 |
| ee | 55 | 21 |

| SELECT DISTINCT * FROM t_1
```





SQL: REMOVE Duplicate Lines - Example #57 - Q: item id val aa 11 17 aa 11 19 bb 22 20 ee 55 21 SELECT item,id, MAX(val) val FROM t_1 GROUP BY item, id



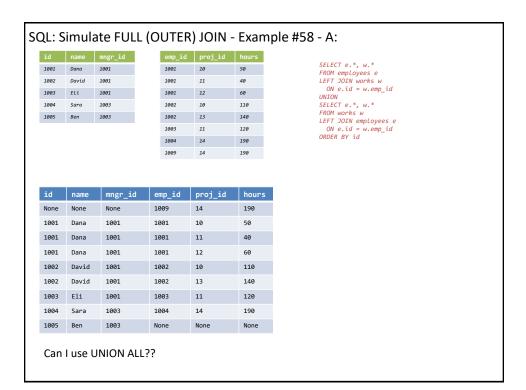
SIMULATE FULL (OUTER) JOIN

SQL: Simulate FULL (OUTER) JOIN - Example #58 - Q:

id	name	mngr_id
1001	Dana	1001
1002	David	1001
1003	Eli	1001
1004	Sara	1003
1005	Ben	1003

emp_id	proj_id	hours
1001	10	50
1001	11	40
1001	12	60
1002	10	110
1002	13	140
1003	11	120
1004	14	190
1009	14	190

```
SELECT e.*, w.*
FROM employees e
LEFT JOIN works w
ON e.id = w.emp_id
UNION
SELECT e.*, w.*
FROM works w
LEFT JOIN employees e
ON e.id = w.emp_id
ORDER BY id
```



Learn SQL by Example (Chananel Perel 2023)

LEFT JOIN - ON VS WHERE

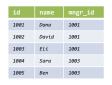
SQL: LEFT JOIN - ON vs WHERE - Example #59 - Q:

id	name	mngr_id	
1001	Dana	1001	
1002	David	1001	
1003	Eli	1001	
1004	Sara	1003	
1005	Ben	1003	

emp_id	proj_id	hours
1001	10	50
1001	11	40
1001	12	60
1002	10	110
1002	13	140
1003	11	120
1004	14	190
1009	14	190

SELECT e.*, w.*
FROM employees e
LEFT JOIN works w
 ON e.id = w.emp_id
 AND proj_id > 10

SQL: LEFT JOIN - ON vs WHERE - Example #59 - A:



emp_id	proj_id	hours
1001	10	50
1001	11	40
1001	12	60
1002	10	110
1002	13	140
1003	11	120
1004	14	190
1009	14	190

SELECT e.*, w.*
FROM employees e
LEFT JOIN works w
ON e.id = w.emp_id
AND proj_id > 10

id	name	mngr_id	emp_id	proj_id	hours
1001	Dana	1001	1001	11	40
1001	Dana	1001	1001	12	60
1002	David	1001	1002	13	140
1003	Eli	1001	1003	11	120
1004	Sara	1003	1004	14	190
1005	Ben	1003	None	None	None

ON runs as part of the JOIN

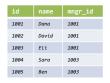
SQL: LEFT JOIN - ON vs WHERE - Example #60 - Q:

name	mngr_id	
Dana	1001	
David	1001	
Eli	1001	
Sara	1003	
Ben	1003	
	Dana David Eli Sara	

emp_id	proj_id	hours
1001	10	50
1001	11	40
1001	12	60
1002	10	110
1002	13	140
1003	11	120
1004	14	190
1009	14	190

SELECT e.*, w.*
FROM employees e
LEFT JOIN works w
ON e.id = w.emp_id
WHERE proj_id > 10

SQL: LEFT JOIN - ON vs WHERE - Example #60 - A:



emp_id	proj_id	hours
1001	10	50
1001	11	40
1001	12	60
1002	10	110
1002	13	140
1003	11	120
1004	14	190
1009	14	190

SELECT e.*, w.*
FROM employees e
LEFT JOIN works w
ON e.id = w.emp_id
WHERE proj_id > 10

id	name	mngr_id	emp_id	proj_id	hours
1001	Dana	1001	1001	11	40
1001	Dana	1001	1001	12	60
1002	David	1001	1002	13	140
1003	Eli	1001	1003	11	120
1004	Sara	1003	1004	14	190

WHERE runs after the JOIN