

Database final project

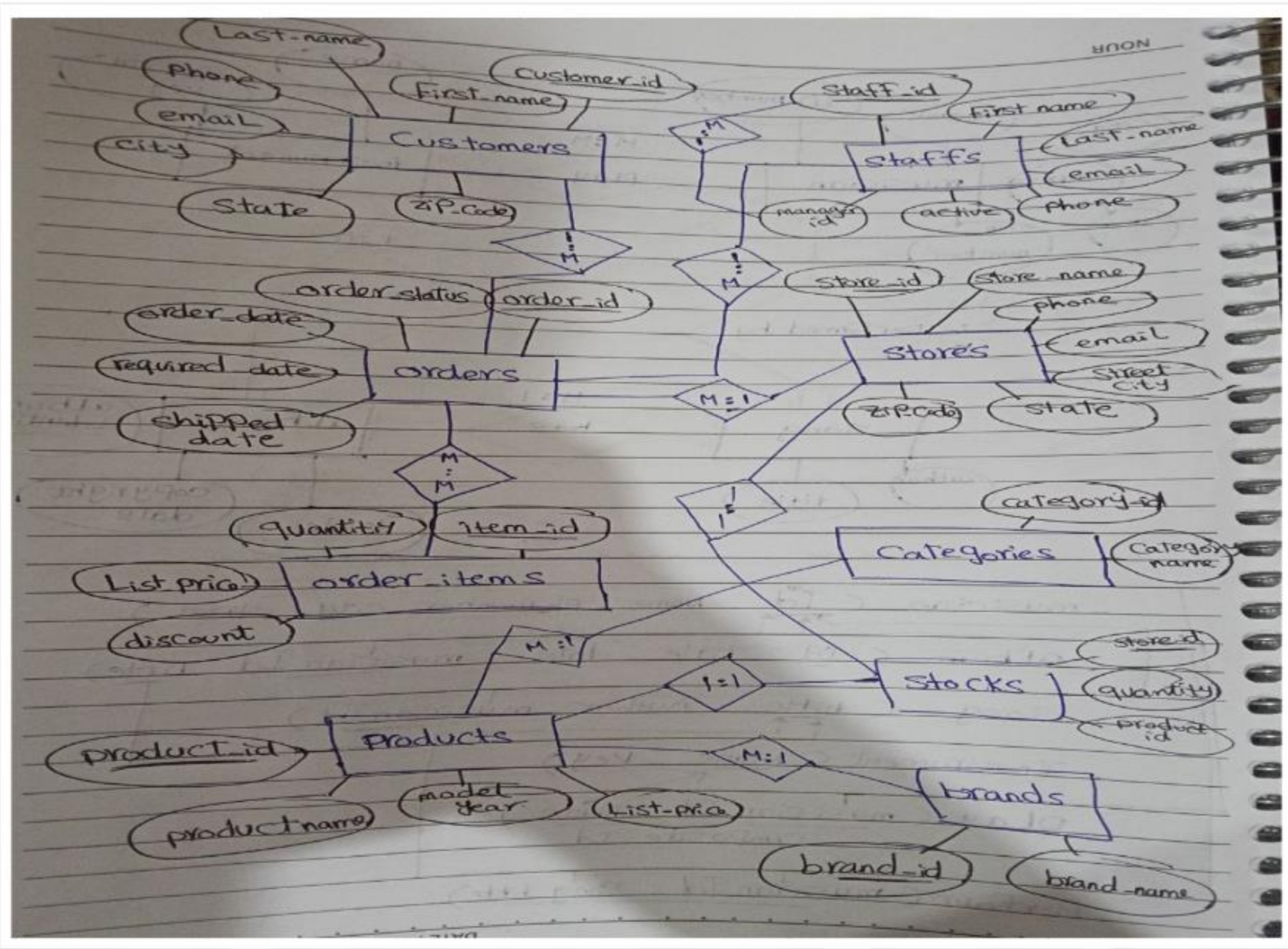
Prepared by : Lobna Saad

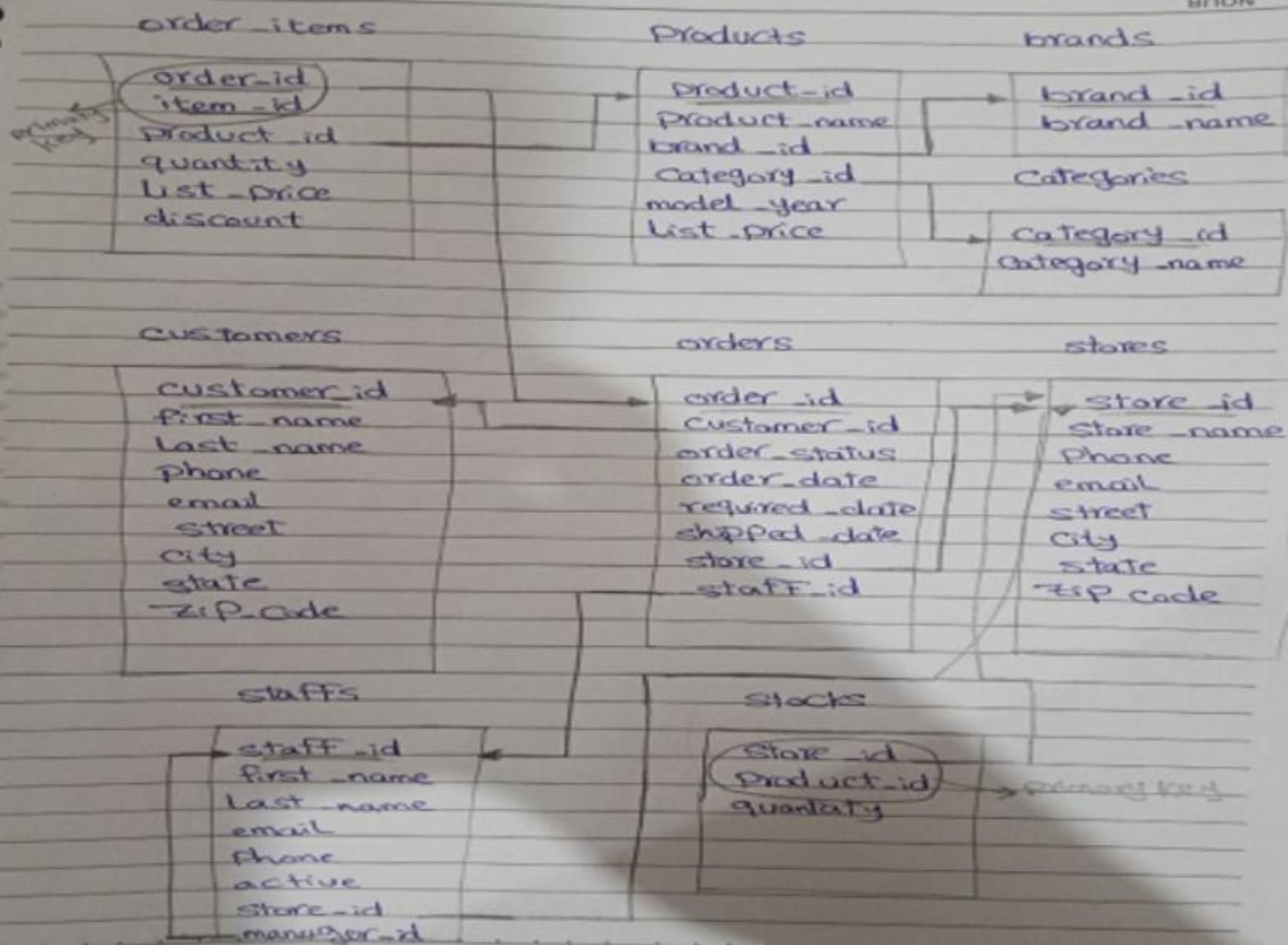


We have a sales datasets to work on:

- ❖ 9 tables
- ❖ We will display the ER diagram and Mapping Schema in the next slide
- ❖ Then display the whole **SQL** code **queries & joins** and its code **run**

Let's start >>





Firstly :

We create a database and name it mutli_datasets and use it.

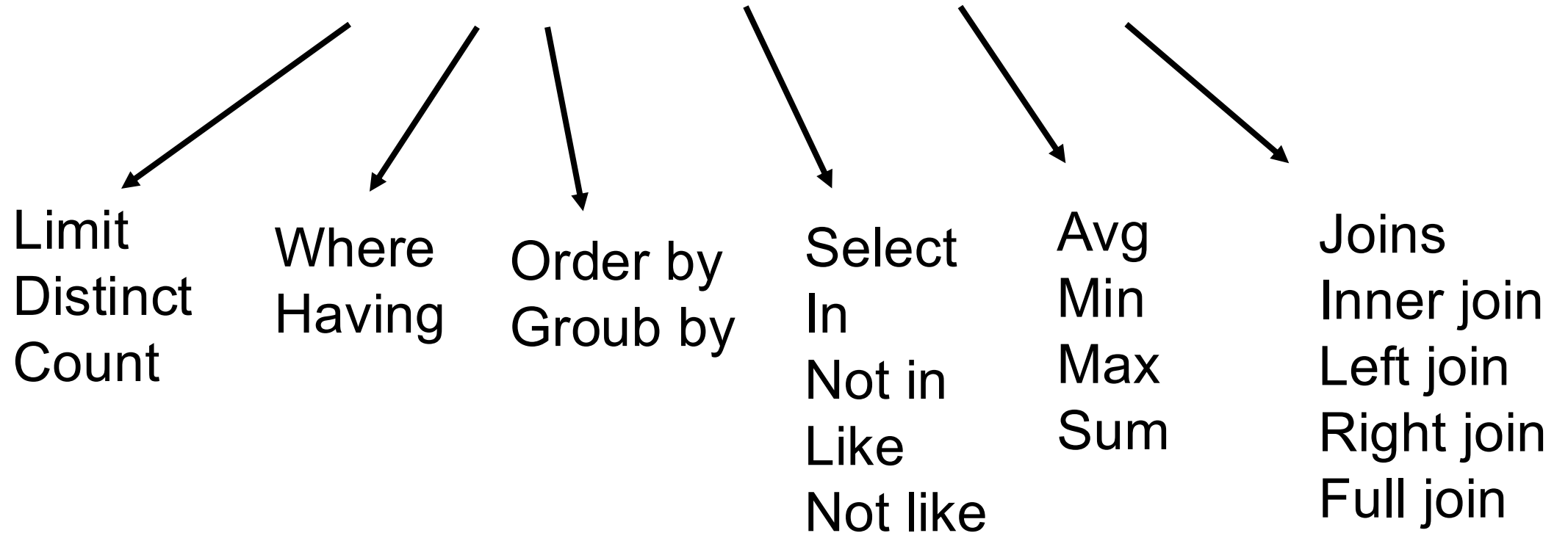
```
create database multi_datasets;  
use multi_datasets;
```

Second :

then we create tables and import data into them.

and them made multiple & different queries, let see it and its code run!!

Select and display data with different conditions



First we will create tables and import data into tables

```
/
8 • create table brands (
9   brand_id int primary key ,
10  brand_name varchar(100)
11 );
12
13 • create table categorise (
14   category_id int primary key,
15   category_name varchar(100)
16 );
17
18 • create table customers (
19   customer_id int primary key ,
20   first_name varchar(100),
21   last_name varchar(100),
22   phone varchar(20) ,
23   email varchar(90),
24   street varchar(50),
25   city varchar(50),
26   state varchar(50),
27   zip_code varchar(10)
28 );
```

```
30 • create table stores (
31   store_id int primary key,
32   store_name varchar(100),
33   phone varchar(20) ,
34   email varchar(50),
35   street varchar(90),
36   city varchar(50),
37   state varchar(10),
38   zip_code varchar(10)
39 );
40
41 • create table products (
42   product_id int primary key,
43   product_name varchar(100),
44   brand_id int ,
45   category_id int,
46   model_year int,
47   list_price decimal(10,2) ,
48   foreign key (brand_id) references brands (brand_id),
49   foreign key (category_id) references categorise (category_id)
50 );
```

```
54 • create table stocks (  
55     store_id int ,  
56     product_id int ,  
57     quantity int,  
58     primary key (store_id , product_id),  
59     foreign key (product_id) references products (product_id),  
60     foreign key (store_id) references stores (store_id)  
61 );  
62  
63 • create table staffs (  
64     staff_id int primary key,  
65     first_name varchar(100),  
66     last_name varchar(100),  
67     email varchar(60),  
68     phone varchar(20),  
69     active int ,  
70     store_id int,  
71     manager_id int,  
72     foreign key (manager_id) references staffs (staff_id),  
73     foreign key (store_id) references stores (store_id)  
-- ,
```

```
77 • create table orders (  
78     order_id int primary key ,  
79     customer_id int ,  
80     order_status int ,  
81     order_date date,  
82     required_date date,  
83     shipped_date date,  
84     store_id int ,  
85     staff_id int ,  
86     foreign key (customer_id) references customers (customer_id),  
87     foreign key (store_id) references stores (store_id),  
88     foreign key (staff_id) references staffs (staff_id)  
89 );
```



```

• create table order_items (
  order_id int ,
  item_id int ,
  product_id int ,
  quantity int ,
  list_price decimal(10,2) ,
  discount decimal(4,2) ,
  primary key (order_id, item_id),
  foreign key (order_id) references orders (order_id),
  foreign key (product_id) references products (product_id)
);

```

```

105 • select *
106       from multi_datasets.brands limit 5;
107
108
109 • select *
110       from multi_datasets.categorise limit 5;
111
112
113 • select *
114       from multi_datasets.customers limit 10;
115
116
117
118 • select *
119       from multi_datasets.order_items limit 10;

```

	order_id	item_id	product_id	quantity	list_price	discount
▶	1	1	20	1	599.99	0.20
	1	2	8	2	1799.99	0.07
	1	3	10	2	1549.00	0.05
	1	4	16	2	599.99	0.05
	1	5	4	1	2899.99	0.20
	2	1	20	1	599.99	0.07
	2	2	16	2	599.99	0.05
	3	1	3	1	999.99	0.05
	3	2	20	1	599.99	0.05
	4	1	2	2	749.99	0.10
-	NULL	NULL	NULL	NULL	NULL	NULL

```
-- *****  
  
-- APPLY DML  
  
-- 1- The SQL INSERT Statement  
-- 2- The SQL UPDATE Statement  
-- 3- The SQL DELETE Statement  
  
-- *****
```

```
153 -- *****  
154 -- Inserting Data INTO a New Table  
155 -- *****  
156 ● create table departments  
157 (   
158     dept_number char(4) not null,  
159     dept_name varchar(40) not null  
160 );  
161  
162 ● insert into departments values ('d010', 'Business Analysis');  
163  
164 ● select  
165     *  
166 from  
167     departments  
168 limit 10;
```

```
164 ● select  
165     *  
166 from  
167     departments  
168 limit 10;  
169 -- *****
```

Result Grid



Filter Rows:

	dept_number	dept_name
▶	d010	Business Analysis

```
select
    *
from
    brands
limit 10;
```

brand_id	brand_name
1	Electra
2	Haro
3	Heller
4	Pure Cycles
5	Ritchey
6	Strider
7	Sun Bicycles
8	Surly
9	Trek
30	lobnaa
NULL	NULL

```
select
    *
from
    categorise
order by category_id desc
limit 10;
```

category_id	category_name
7	Road Bikes
6	Mountain Bikes
5	Electric Bikes
4	Cyclocross Bicycles
3	Cruisers Bicycles
2	Comfort Bicycles
1	Children Bicycles
NULL	NULL

```
SELECT
    *
FROM
    products
ORDER BY model_year;

DELETE FROM products;

ROLLBACK;
```

product_id	product_name	brand_id	category_id	model_year	list_price
1	Trek 820 - 2016	9	6	2016	379.99
2	Ritchey Timberwolf Frameset - 2016	5	6	2016	749.99
3	Surly Wednesday Frameset - 2016	8	6	2016	999.99
4	Trek Fuel EX 8 29 - 2016	9	6	2016	2899.99
5	Heller Shagmaw Frame - 2016	3	6	2016	1320.99
6	Surly Ice Cream Truck Frameset - 2016	8	6	2016	469.99
7	Trek Slash 8 27.5 - 2016	9	6	2016	3999.99
8	Trek Remedy 29 Carbon Frameset - 2016	9	6	2016	1799.99
9	Trek Conduit+ - 2016	9	5	2016	2999.99
10	Surly Straggler - 2016	8	4	2016	1549.00
11	Surly Straggler 650b - 2016	8	4	2016	1680.99
12	Electra Townie Original 21D - 2016	1	3	2016	549.99
13	Electra Cruiser 1 (24-Inch) - 2016	1	3	2016	269.99

```
-- *****
-- update sql statment
-- *****
```

```
173 • select * from customers
174 where
175     customer_id = 30;
```

	customer_id	first_name	last_name	phone	email	street	city	state	zip_code
▶	30	lobnaa	saad	NULL	jamaal.albert@gmail.com	853 Stonybrook Street	kafrelsheikh	CA	90505
⬇	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
177 • update customers
178 set
179     first_name = 'lobnaa',
180     last_name = 'saad',
181     city = 'kafrelsheikh'
182 where
183     customer_id = 30;
184 • select *
185 from customers
```

<div> <div>Result Grid</div> <div> Filter Rows: <input type="text"/> </div> <div> <div>Edit:</div> <div> </div> <div>Export/Import:</div> <div> </div> <div>Wrap Cell Cont</div> </div> </div>									
	customer_id	first_name	last_name	phone	email	street			
	27	Santos	Valencia	NULL	santos.valencia@yahoo.com	7479 Carpenter Street			
	28	Jeanice	Frost	NULL	jeanice.frost@hotmail.com	76 Devon Lane			
	29	Syreeta	Hendricks	NULL	syreeta.hendricks@msn.com	193 Spruce Road			
	30	lobnaa	saad	NULL	jamaal.albert@gmail.com	853 Stonybrook Street			


```
-- *****
-- APPLY DQL
-- The SQL SELECT Statement
-- AND & OR & NOT
-- in & not in
-- like & not like
-- between and
-- is null & is not null
-- other comparison operators
-- select distinct >> 'use to return unique values'
-- *****
```

```
229 • SELECT
230     staff_id, first_name, last_name, active,
231     store_id, manager_id
232 FROM
233     staffs
234 WHERE
235     first_name = 'Kali';
236
```

237 *****

Result Grid |   Filter Rows: | Edit:   

	staff_id	first_name	last_name	active	store_id	manager_id
▶	8	Kali	Vargas	1	3	1
⌵	NULL	NULL	NULL	NULL	NULL	NULL


```

238 • SELECT
239     staff_id, first_name, last_name, active,
240     store_id, manager_id
241 FROM
242     staffs
243 WHERE
244     first_name = 'Kali' AND active = 1; -- A

```

Result Grid | Filter Rows: | Edit: |

	staff_id	first_name	last_name	active	store_id	manager_id
▶	8	Kali	Vargas	1	3	1
•	NULL	NULL	NULL	NULL	NULL	NULL

```

286 • SELECT
287     *
288 FROM
289     customers
290 WHERE
291     first_name IN ('Tameka', 'Kasha', 'Kaylee');

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

	customer_id	first_name	last_name	phone	email	street	city	state	zip_code
▶	2	Kasha	Todd	NULL	kasha.todd@yahoo.com	910 Vine Street	Campbell	CA	95008
	3	Tameka	Fisher	NULL	tameka.fisher@aol.com	769C Honey Creek St.	Redondo Beach	CA	90278
	23	Kaylee	English	NULL	kaylee.english@msn.com	8786 Fulton Rd.	Hollis	NY	11423
	727	Kasha	Sullivan	NULL	kasha.sullivan@hotmail.com	48 Foster Rd.	Scarsdale	NY	10583
•	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```

282 • SELECT
283     customer_id, first_name, last_name, city, state
284 FROM
285     customers
286 WHERE
287     first_name = 'Tameka'
288     OR first_name = 'Kasha'
289     OR first_name = 'Kaylee';

```

Result Grid | Filter Rows: | Edit: |

	customer_id	first_name	last_name	city	state
▶	2	Kasha	Todd	Campbell	CA
	3	Tameka	Fisher	Redondo Beach	CA
	23	Kaylee	English	Hollis	NY
	727	Kasha	Sullivan	Scarsdale	NY
•	NULL	NULL	NULL	NULL	NULL

```

299 SELECT
300     *
301 FROM
302     customers
303 WHERE
304     first_name not IN ('Tameka' , 'Kasha', 'Kaylee');
305
306 -- .....
307

```

Result Grid Filter Rows: Edit: Export/Import: Wrap Cell Content: Fetch rows:									
	customer_id	first_name	last_name	phone	email	street	city	state	zip_code
▶	1	Debra	Burks	NULL	debra.burks@yahoo.com	9273 Thorne Ave.	Orchard Park	NY	14127
	4	Daryl	Spence	NULL	daryl.spence@aol.com	988 Pearl Lane	Uniondale	NY	11553
	5	Charolette	Rice	(916) 381-6003	charolette.rice@msn.com	107 River Dr.	Sacramento	CA	95820
	6	Lyndsey	Bean	NULL	lyndsey.bean@hotmail.com	769 West Road	Fairport	NY	14450
	7	Latasha	Hays	(716) 986-3359	latasha.hays@hotmail.com	7014 Manor Station Rd.	Buffalo	NY	14215
	8	Jacqueline	Duncan	NULL	jacqueline.duncan@yahoo.com	15 Brown St.	Jackson Heights	NY	11372
	9	Genoveva	Baldwin	NULL	genoveva.baldwin@msn.com	8550 Spruce Drive	Port Washington	NY	11050
	10	Pamela	Newman	NULL	pamela.newman@gmail.com	476 Chestnut Ave.	Monroe	NY	10950
	11	Deshawn	Mendoza	NULL	deshawn.mendoza@yahoo.com	8790 Cobblestone Street	Monsey	NY	10952
	12	Robby	Sykes	(516) 583-7761	robby.sykes@hotmail.com	486 Rock Maple Street	Hempstead	NY	11550
	13	Lashawn	Ortiz	NULL	lashawn.ortiz@msn.com	27 Washington Rd.	Longview	TX	75604
	14	Garry	Espinoza	NULL	garry.espinoza@hotmail.com	7858 Rockaway Court	Forney	TX	75126
	15	Linnie	Branch	NULL	linnie.branch@gmail.com	314 South Columbia Ave.	Plattsburgh	NY	12901

customers 17 x

```
260 • SELECT
261      *
262  FROM
263      customers
264  WHERE
265      first_name = 'Kasha' AND last_name = 'Todd' or customer_id = 3;
266
```

[illegible]

```
250      -- unexpected syntax
251
252 •    SELECT
253         *
254     FROM
255         customers
256     WHERE
257         first_name = 'Debra' AND first_name = 'Kasha';
258
```

[illegible]

```

268 • SELECT
269     *
270 FROM
271     customers
272 WHERE
273     first_name = 'Kasha' AND (last_name = 'Todd' or customer_id = 3);
274

```

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

	customer_id	first_name	last_name	phone	email	street	city	state	zip_code
▶	2	Kasha	Todd	NULL	kasha.todd@yahoo.com	910 Vine Street	Campbell	CA	95008
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```

308 • SELECT
309     first_name, last_name, customer_id
310 FROM
311     customers
312 WHERE
313     first_name LIKE('S%'); -- name starts with S
314     -- *****
315

```

316 • SELECT

Result Grid | Filter Rows: | Edit:

first_name	last_name	customer_id
Santos	Valencia	27
Syreeta	Hendricks	29
Saturnina	Garner	53
Sarai	McKee	80
Sharyn	Hopkins	94
Sylvie	Wilkerson	103
Shae	Hickman	108
Shila	White	111
Shena	Carter	122

```

386 • SELECT
387     COUNT(manager_id)
388 FROM
389     staffs;

```

Result Grid | Filter Rows:

COUNT(manager_id)
9

```

391 • SELECT
392     COUNT(DISTINCT manager_id)
393 FROM
394     staffs;

```

Result Grid | Filter Rows: | Export:

COUNT(DISTINCT manager_id)
4


```

316 • SELECT
317     first_name, last_name, customer_id
318 FROM
319     customers
320 WHERE
321     first_name LIKE('%s%'); -- name contain s
322     *****

```

Result Grid



Filter Rows:

Edit:



	first_name	last_name	customer_id
▶	Kasha	Todd	2
	Lyndsey	Bean	6
	Latasha	Hays	7
	Deshawn	Mendoza	11
	Lashawn	Ortiz	13
	Santos	Valencia	27
	Syreeta	Hendricks	29
	Deloris	Burke	33
	Cesar	Jackson	48
	Saturnina	Garner	53
	Latasha	Stanley	59

```

323 • SELECT
324     first_name, last_name, customer_id
325 FROM
326     customers
327 WHERE
328     first_name NOT LIKE ('%s%');
329     -- *****

```

Result Grid



Filter Rows:

Edit:



	first_name	last_name	customer_id
▶	Debra	Burks	1
	Tameka	Fisher	3
	Daryl	Spence	4
	Charolette	Rice	5
	Jacqueline	Duncan	8
	Genoveva	Baldwin	9
	Pamelia	Newman	10
	Robby	Sykes	12
	Garry	Espinoza	14
	Linnie	Branch	15
	Emmitt	Sanchez	16
	Caren	Stephens	17
	Georgetta	Hardin	18
	Lizzette	Stein	19


```

330 • SELECT
331     model_year
332 FROM
333     products
334 WHERE
335     model_year BETWEEN '2016' AND '2018';
336 -- *****

```

Result Grid



Filter Rows:

Export:



model_year
2016
2016
2016
2016
2017
2017
2017
2017

```

345 • SELECT
346     manager_id
347 FROM
348     staffs
349 WHERE
350     manager_id IS NULL;
351 -- *****

```

Result Grid



Filter Rows:

manager_id
NULL

```

352 • SELECT
353     manager_id
354 FROM
355     staffs
356 WHERE
357     manager_id <> 'Null';
358 -- same above code != dont equal mark

```

Result Grid   Filter Rows: Export

	manager_id
▶	1
	1
	1
	2
	2
	5
	5
	7
	7

```

358 -- same above code := dont equal mark
359 • SELECT
360     manager_id
361 FROM
362     staffs
363 WHERE
364     manager_id != 'Null';
365 -- *****

```

Result Grid   Filter Rows: Export

	manager_id
▶	1
	1
	1
	2
	2
	5
	5
	7
	7

```
366 SELECT DISTINCT
367     manager_id
368 FROM
369     staffs;
370
```

Result Grid   Filter Rows



	manager_id
▶	NULL
	1
	2
	5
	7

```
-- *****
-- MySQL Aggregation Functions
-- count() >> count rows
-- sum()
-- min()
-- max()
-- avg()
-- round()
-- group by()
-- order by()
-- having()
-- *****
```

```

407 • SELECT
408     SUM(manager_id)
409 FROM
410     staffs;
411 -- *****

```



Result Grid |   Filter Rows:

SUM(manager_id)
31

```

---
412 • SELECT
413     MAX(manager_id)
414 FROM
415     staffs;

```



Result Grid |   Filter Rows:

SUM(manager_id)
31

```

410
417 • SELECT
418     MIN(manager_id)
419 FROM
420     staffs;
421 -- *****

```

Result Grid |   Filter Rows:

MIN(manager_id)
1

```

---
422 • SELECT
423     AVG(manager_id)
424 FROM
425     staffs;
426 -- *****

```



Result Grid |   Filter Rows:

AVG(manager_id)
3.4444

```

427 • SELECT
428     ROUND(AVG(manager_id))
429 FROM
430     staffs;
431

```

Result Grid |   Filter Rows:

ROUND(AVG(manager_id))
3

```

432 • SELECT
433     ROUND(AVG(manager_id), 2)
434 FROM
435     staffs;
436 -- *****

```

Result Grid |   Filter Rows:

ROUND(AVG(manager_id), 2)
3.44

```

431 • SELECT
432     *
433 FROM
434     brands
435 ORDER BY brand_id desc;

```

Result Grid   Filter Rows:

	brand_id	brand_name
▶	31	lobnaa
	30	lobnaa
	9	Trek
	8	Surly
	7	Sun Bicydes
	6	Strider
	5	Ritchey
	4	Pure Cycles
	3	Heller
	2	Haro
	1	Electra
•	NULL	NULL

```

437 • SELECT
438     brand_id
439 FROM
440     brands
441 GROUP BY brand_id;

```

Result Grid   Filter Rows:

	brand_id
▶	1
	2
	3
	4
	5
	6
	7
	8
	9
	30
	31
•	NULL

```

332 • SELECT
333     manager_id
334 FROM
335     staffs
336 WHERE
337     manager_id IS NOT NULL;

```

Result Grid   Filter Rows: Exp


	manager_id
▶	1
	1
	1
	2
	2
	5
	5
	7
	7


```
-- *****  
  
-- joins  
-- 1- inner join  
-- 2-left join  
-- 3-right join  
-- 4-full join  
-- 5-self join  
-- *****
```

```

468 • select first_name,last_name,email,order_id,order_status, order_date
469      from orders O, customers C
470      where C.customer_id = O.customer_id
471      order by O.customer_id desc;
472

```

Result Grid |  Filter Rows: | Export:  | Wrap Cell Content:  | Fetch rows: 

	first_name	last_name	email	order_id	order_status	order_date
▶	Ester	Acevedo	ester.acevedo@gmail.com	1424	4	2018-03-07
	Ivette	Estes	ivette.estes@gmail.com	616	4	2016-12-20
	Lezlie	Lamb	lezlie.lamb@gmail.com	558	4	2016-11-15
	Cassie	Cline	cassie.cline@gmail.com	1036	4	2017-07-30
	Jamaal	Morrison	jamaal.morrison@msn.com	1366	4	2018-01-26
	Ernest	Rollins	ernest.rollins@yahoo.com	466	4	2016-09-28
	Florrie	Little	florrie.little@yahoo.com	1053	4	2017-08-10
	Lee	Dunn	lee.dunn@aol.com	927	4	2017-06-03
	Son	Warner	son.warner@hotmail.com	397	4	2016-08-28
	Macie	Ayers	macie.ayers@msn.com	844	4	2017-04-16
	Merrie	Fowler	merrie.fowler@aol.com	234	4	2016-05-22
	Elana	Miles	elana.miles@yahoo.com	43	4	2016-01-27
	Nicola	Knight	nicola.knight@aol.com	1352	4	2018-01-16
	Zona	Cameron	zona.cameron@aol.com	963	4	2017-06-19

```

490 • select first_name , last_name, order_id, order_date, store_name, s.street, s.city
491   from customers c, orders o, stores s
492   where c.customer_id = o.customer_id and o.store_id = s.store_id;
493
494   -- same code
495
496 • select first_name , last_name, order_id, order_date, store_name, s.street, s.city
497   from customers c join orders o on c.customer_id = o.customer_id join stores s on o.store_id = s.store_id;
498   -- *****

```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

Fetch rows:

	first_name	last_name	order_id	order_date	store_name	street	city
▶	Johnathan	Velazquez	1	2016-01-01	Santa Cruz Bikes	3700 Portola Drive	Santa Cruz
	Nova	Hess	4	2016-01-03	Santa Cruz Bikes	3700 Portola Drive	Santa Cruz
	Neil	Mccall	9	2016-01-05	Santa Cruz Bikes	3700 Portola Drive	Santa Cruz
	Marvin	Mullins	12	2016-01-06	Santa Cruz Bikes	3700 Portola Drive	Santa Cruz
	Maribel	William	14	2016-01-09	Santa Cruz Bikes	3700 Portola Drive	Santa Cruz
	Lea	Key	16	2016-01-12	Santa Cruz Bikes	3700 Portola Drive	Santa Cruz
	Sindy	Anderson	17	2016-01-12	Santa Cruz Bikes	3700 Portola Drive	Santa Cruz
	Lanita	Burton	18	2016-01-14	Santa Cruz Bikes	3700 Portola Drive	Santa Cruz






Activate Windows

Go to Settings to activate Windows.

```

506  -- *****
507  -- left join
508  • select first_name, last_name , order_id,order_date , shipped_date
509      from customers c
510      left join orders o on c.customer_id = o.customer_id;
511  -- *****

```




Result Grid |   Filter Rows: | Export:  | Wrap Cell Content:  | Fetch rows: 

	first_name	last_name	order_id	order_date	shipped_date
▶	Debra	Burks	599	2016-12-09	2016-12-12
	Debra	Burks	1555	2018-04-18	NULL
	Debra	Burks	1613	2018-11-18	NULL
	Kasha	Todd	692	2017-02-05	NULL
	Kasha	Todd	1084	2017-08-21	2017-08-23
	Kasha	Todd	1509	2018-04-09	NULL
	Tameka	Fisher	1468	2018-03-27	2018-03-29
	Tameka	Fisher	1496	2018-04-06	NULL
	Tameka	Fisher	1612	2018-10-21	NULL
	Daryl	Spence	700	2017-02-07	2017-02-08
	Daryl	Spence	1259	2017-11-21	NULL
	Daryl	Spence	1556	2018-04-18	NULL
	Charolette	Rice	264	2016-06-10	NULL
	Charolette	Rice	571	2016-11-24	2016-11-27


```

512  -- right join
513 •  select first_name, last_name, order_id, order_date, shipped_date
514      from customers c
515      right join orders o on c.customer_id = o.customer_id;
516  ****

```

Result Grid |   Filter Rows: | Export:  | Wrap Cell Content:  | Fetch rows: 

	first_name	last_name	order_id	order_date	shipped_date
▶	Johnathan	Velazquez	1	2016-01-01	2016-01-03
	Jaqueline	Cummings	2	2016-01-01	2016-01-03
	Joshua	Robertson	3	2016-01-02	2016-01-03
	Nova	Hess	4	2016-01-03	2016-01-05
	Arla	Ellis	5	2016-01-03	2016-01-06
	Sharyn	Hopkins	6	2016-01-04	2016-01-05
	Laureen	Paul	7	2016-01-04	2016-01-05
	Leslie	Higgins	8	2016-01-04	2016-01-05
	Neil	Mccall	9	2016-01-05	2016-01-08
	Alane	Munoz	10	2016-01-05	2016-01-06
	Tarra	Guerrero	11	2016-01-05	2016-01-07
	Marvin	Mullins	12	2016-01-06	2016-01-09
	Patience	Clayton	13	2016-01-08	2016-01-11
	Maribel	William	14	2016-01-09	2016-01-12
	Ellsworth	Michael	15	2016-01-09	2016-01-12
	Lea	Key	16	2016-01-12	2016-01-15
	Sinda	Anderson	17	2016-01-12	2016-01-14


```

517  -- full join
518 •  select first_name, last_name,order_id,order_date, shipped_date
519      from customers c
520      left join orders o on c.customer_id = o.customer_id
521
522      union
523
524      select first_name, last_name,order_id,order_date, shipped_date
525      from customers c
526      right join orders o on c.customer_id = o.customer_id;
527

```

Result Grid |  Filter Rows: | Export:  | Wrap Cell Content: 

	first_name	last_name	order_id	order_date	shipped_date
►	Debra	Burks	599	2016-12-09	2016-12-12
	Debra	Burks	1555	2018-04-18	NULL
	Debra	Burks	1613	2018-11-18	NULL
	Kasha	Todd	692	2017-02-05	NULL
	Kasha	Todd	1084	2017-08-21	2017-08-23
	Kasha	Todd	1509	2018-04-09	NULL
	Tameka	Fisher	1468	2018-03-27	2018-03-29
	Tameka	Fisher	1496	2018-04-06	NULL

```

528      -- self join
529 •   select c.first_name , c.last_name, u.first_name, u.last_name
530      from customers c
531      right join customers u on c.customer_id = u.customer_id;
532

```

Result Grid   Filter Rows:

Export:  Wrap Cell Content:  Fetch rows: 

	first_name	last_name	first_name	last_name
▶	Debra	Burks	Debra	Burks
	Kasha	Todd	Kasha	Todd
	Tameka	Fisher	Tameka	Fisher
	Daryl	Spence	Daryl	Spence
	Charolette	Rice	Charolette	Rice
	Lyndsey	Bean	Lyndsey	Bean
	Latasha	Hays	Latasha	Hays
	Jacqueline	Duncan	Jacqueline	Duncan
	Genoveva	Baldwin	Genoveva	Baldwin
	Pamelia	Newman	Pamelia	Newman
	Deshawn	Mendoza	Deshawn	Mendoza
	Robby	Sykes	Robby	Sykes
	Lashawn	Ortiz	Lashawn	Ortiz
	Garry	Espinoza	Garry	Espinoza
	Lionie	Branch	Lionie	Branch



Any questions?

Thanks