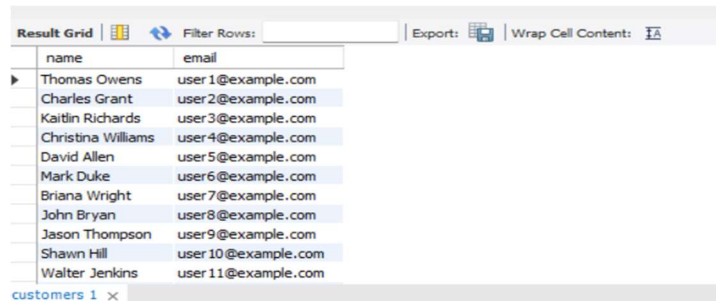


LEVEL 1

1. Retrieve customer names and emails for email marketing

select name ,email from customers;



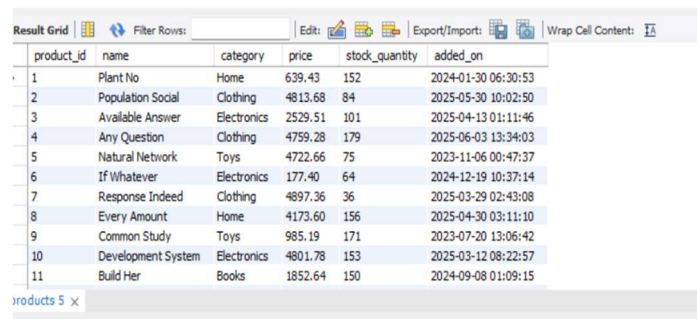
The screenshot shows a database query result grid with two columns: 'name' and 'email'. The grid contains 11 rows of customer data. The interface includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Content' toggle.

name	email
Thomas Owens	user1@example.com
Charles Grant	user2@example.com
Kaitlin Richards	user3@example.com
Christina Williams	user4@example.com
David Allen	user5@example.com
Mark Duke	user6@example.com
Briana Wright	user7@example.com
John Bryan	user8@example.com
Jason Thompson	user9@example.com
Shawn Hill	user10@example.com
Walter Jenkins	user11@example.com

customers 1 x

2. View complete product catalog with all available details

select * from products;



The screenshot shows a database query result grid with six columns: 'product_id', 'name', 'category', 'price', 'stock_quantity', and 'added_on'. The grid contains 11 rows of product data. The interface includes a 'Filter Rows' field, an 'Edit' button, an 'Export/Import' button, and a 'Wrap Cell Content' toggle.

product_id	name	category	price	stock_quantity	added_on
1	Plant No	Home	639.43	152	2024-01-30 06:30:53
2	Population Social	Clothing	4813.68	84	2025-05-30 10:02:50
3	Available Answer	Electronics	2529.51	101	2025-04-13 01:11:46
4	Any Question	Clothing	4759.28	179	2025-06-03 13:34:03
5	Natural Network	Toys	4722.66	75	2023-11-06 00:47:37
6	If Whatever	Electronics	177.40	64	2024-12-19 10:37:14
7	Response Indeed	Clothing	4897.36	36	2025-03-29 02:43:08
8	Every Amount	Home	4173.60	156	2025-04-30 03:11:10
9	Common Study	Toys	985.19	171	2023-07-20 13:06:42
10	Development System	Electronics	4801.78	153	2025-03-12 08:22:57
11	Build Her	Books	1852.64	150	2024-09-08 01:09:15

products 5 x

3. List all unique product categories

`select distinct category from products;`

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

category
Home
Clothing
Electronics
Toys
Books

products 5 x

4. Show all products priced above ₹1,000

`select * from products where price>1000;`

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

product_id	name	category	price	stock_quantity	added_on
2	Population Social	Clothing	4813.68	84	2025-05-30 10:02:50
3	Available Answer	Electronics	2529.51	101	2025-04-13 01:11:46
4	Any Question	Clothing	4759.28	179	2025-06-03 13:34:03
5	Natural Network	Toys	4722.66	75	2023-11-06 00:47:37
7	Response Indeed	Clothing	4897.36	36	2025-03-29 02:43:08
8	Every Amount	Home	4173.60	156	2025-04-30 03:11:10
10	Development System	Electronics	4801.78	153	2025-03-12 08:22:57
11	Build Her	Books	1852.64	150	2024-09-08 01:09:15
12	Action Ask	Electronics	4017.01	19	2025-02-14 03:38:06
13	Full West	Books	2112.33	172	2023-09-15 03:13:38
16	Special Fact	Toys	1094.18	15	2025-03-17 10:42:40

products 7 x

5. Display products within a mid-range price bracket (₹2,000 to ₹5,000)

```
select * from products where price>2000 and price<5000;
select * from products where price between 2000 and 5000;
```

product_id	name	category	price	stock_quantity	added_on
2	Population Social	Clothing	4813.68	84	2025-05-30 10:02:50
3	Available Answer	Electronics	2529.51	101	2025-04-13 01:11:46
4	Any Question	Clothing	4759.28	179	2025-06-03 13:34:03
5	Natural Network	Toys	4722.66	75	2023-11-06 00:47:37
7	Response Indeed	Clothing	4897.36	36	2025-03-29 02:43:08
8	Every Amount	Home	4173.60	156	2025-04-30 03:11:10
10	Development System	Electronics	4801.78	153	2025-03-12 08:22:57
12	Action Ask	Electronics	4017.01	19	2025-02-14 03:38:06
13	Full West	Books	2112.33	172	2023-09-15 03:13:38
17	Everything Plant	Books	2496.68	120	2023-10-08 20:11:55
18	Some Them	Toys	3673.86	110	2024-07-25 00:33:53

products 8 x

6. Fetch data for specific customer IDs (e.g., from loyalty program list)

```
select * from customers where customer_id in(3,4,5);
```

customer_id	name	email	phone	created_at
3	Kaitlin Richards	user3@example.com	2073473421	2024-06-23 09:55:22
4	Christina Williams	user4@example.com	586-605-5061x06	2024-10-27 17:19:38
5	David Allen	user5@example.com	(751)456-8289x1	2023-10-29 02:43:00
*	NULL	NULL	NULL	NULL

customers 9 x

7. Identify customers whose names start with the letter 'A'

`select name from customers where name like 'a%';`

`select * from customers where name like 'a%';`

The screenshot shows a 'Result Grid' window with a table of customer data. The table has columns: customer_id, name, email, phone, and created_at. The data is filtered to show only customers whose names start with 'A'. The table is titled 'customers 11 x'.

customer_id	name	email	phone	created_at
15	Austin Flores	user15@example.com	329.901.1576x66	2024-06-13 09:03:42
16	Amy Landry	user16@example.com	+1-278-019-3748	2024-02-28 17:51:50
19	Amanda Bright	user19@example.com	380.981.9798x69	2024-12-20 22:58:15
27	Adrienne Green	user27@example.com	530.644.8455x93	2023-08-22 01:55:29
NULL	NULL	NULL	NULL	NULL

8. List electronics products priced under ₹3,000

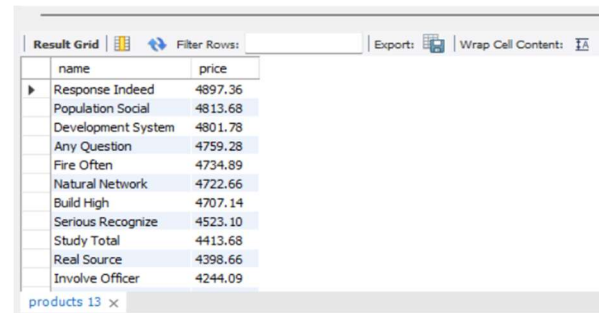
`select * from products where category='electronics' and price<3000;`

The screenshot shows a 'Result Grid' window with a table of product data. The table has columns: product_id, name, category, price, stock_quantity, and added_on. The data is filtered to show only electronics products priced under ₹3,000. The table is titled 'products 12 x'.

product_id	name	category	price	stock_quantity	added_on
3	Available Answer	Electronics	2529.51	101	2025-04-13 01:11:46
6	If Whatever	Electronics	177.40	64	2024-12-19 10:37:14
15	Place Low	Electronics	723.97	46	2023-07-05 14:36:07
31	Series Page	Electronics	2070.37	83	2024-04-01 00:24:06
34	Despite Win	Electronics	1340.34	64	2024-11-27 06:55:45
44	Actually Term	Electronics	396.11	85	2023-11-02 13:09:20
47	Southern Thing	Electronics	512.46	40	2024-02-28 17:57:38
NULL	NULL	NULL	NULL	NULL	NULL

9. Display product names and prices in descending order of price

select name , price from products order by price desc;



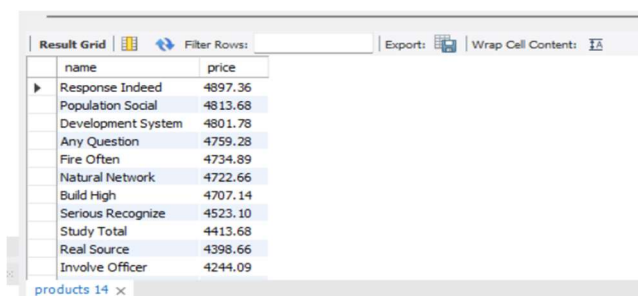
The screenshot shows a 'Result Grid' window with a table of product names and prices. The table is sorted by price in descending order. The interface includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The table has two columns: 'name' and 'price'.

name	price
Response Indeed	4897.36
Population Social	4813.68
Development System	4801.78
Any Question	4759.28
Fire Often	4734.89
Natural Network	4722.66
Build High	4707.14
Serious Recognize	4523.10
Study Total	4413.68
Real Source	4398.66
Involve Officer	4244.09

products 13 x

10. Display product names and prices, sorted by price and then by name

select name, price from products order by price desc , name asc;



The screenshot shows a 'Result Grid' window with a table of product names and prices. The table is sorted by price in descending order, and then by name in ascending order. The interface includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The table has two columns: 'name' and 'price'.

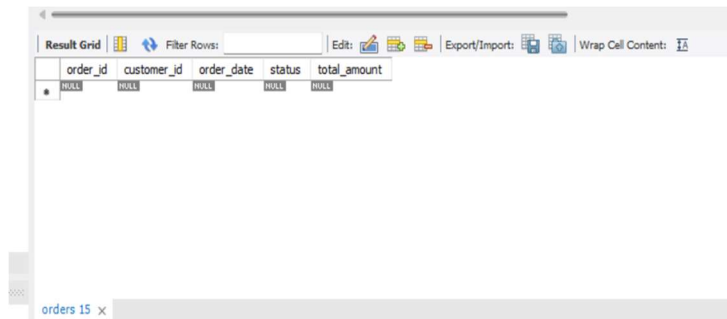
name	price
Response Indeed	4897.36
Population Social	4813.68
Development System	4801.78
Any Question	4759.28
Fire Often	4734.89
Natural Network	4722.66
Build High	4707.14
Serious Recognize	4523.10
Study Total	4413.68
Real Source	4398.66
Involve Officer	4244.09

products 14 x

Level 2

1. Retrieve orders where customer information is missing (possibly due to data migration or deletion)

```
select * from orders where customer_id is null;
```

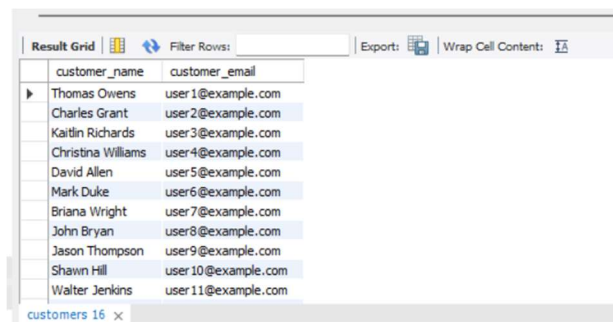


The screenshot shows a database query result grid with the following columns: order_id, customer_id, order_date, status, and total_amount. All cells in the first row are highlighted in red, indicating null values. The grid is titled 'orders 15 x'.

order_id	customer_id	order_date	status	total_amount

2. Display customer names and emails using column aliases for frontend readability

```
select name as customer_name , email as customer_email from customers;
```

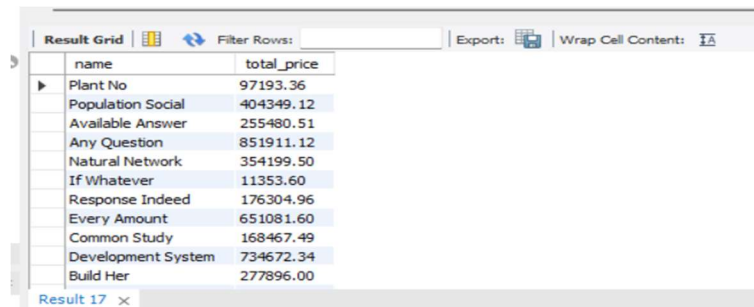


The screenshot shows a database query result grid with the following columns: customer_name and customer_email. The grid displays 16 rows of customer data. The grid is titled 'customers 16 x'.

customer_name	customer_email
Thomas Owens	user1@example.com
Charles Grant	user2@example.com
Kaitlin Richards	user3@example.com
Christina Williams	user4@example.com
David Allen	user5@example.com
Mark Duke	user6@example.com
Briana Wright	user7@example.com
John Bryan	user8@example.com
Jason Thompson	user9@example.com
Shawn Hill	user10@example.com
Walter Jenkins	user11@example.com

3. Calculate total value per item ordered by multiplying quantity and item price

```
select name , price*stock_quantity as total_price from products;
```

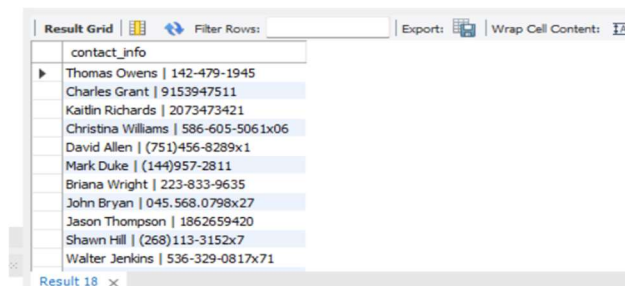


The screenshot shows a 'Result Grid' window with a table containing two columns: 'name' and 'total_price'. The table lists various items and their corresponding total prices calculated from the query.

name	total_price
Plant No	97193.36
Population Social	404349.12
Available Answer	255480.51
Any Question	851911.12
Natural Network	354199.50
If Whatever	11353.60
Response Indeed	176304.96
Every Amount	651081.60
Common Study	168467.49
Development System	734672.34
Build Her	277896.00

4. Combine customer name and phone number in a single column

```
select concat(name," | ",phone) as contact_info from customers;
```



The screenshot shows a 'Result Grid' window with a table containing one column: 'contact_info'. The table lists customer names followed by their phone numbers, separated by a vertical bar and spaces, as concatenated by the query.

contact_info
Thomas Owens 142-479-1945
Charles Grant 9153947511
Kaitlin Richards 2073473421
Christina Williams 586-605-5061x06
David Allen (751)456-8289x1
Mark Duke (144)957-2811
Briana Wright 223-833-9635
John Bryan 045.568.0798x27
Jason Thompson 1862659420
Shawn Hill (268)113-3152x7
Walter Jenkins 536-329-0817x71

5. Extract only the date part from order timestamps for date-wise reporting

```
select date(order_date) from orders;
```

date(order_date)
2025-03-02
2024-10-09
2025-05-08
2024-09-19
2025-04-08
2024-10-25
2024-07-29
2024-07-30
2025-06-10
2025-02-16
2025-03-11

6. List products that do not have any stock left

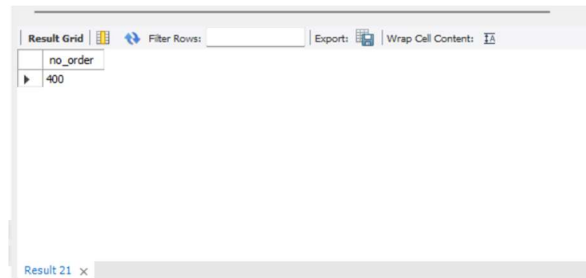
```
select * from products where stock_quantity<0;
```

product_id	name	category	price	stock_quantity	added_on
1	MOUSE	MOUSE	1000	-1	2025-03-11

Level 3

1. Count the total number of orders placed

```
select count(*) as no_order from orders;
```

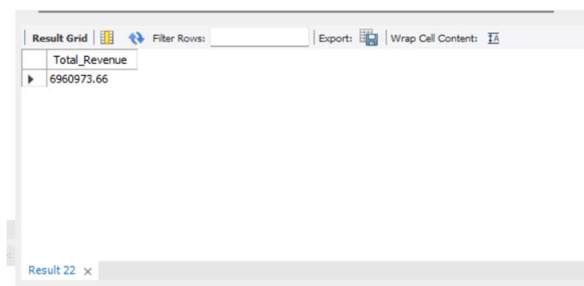


The screenshot shows a database query result grid. The grid has two columns: 'no_order' and a value '400'. The interface includes a 'Result Grid' tab, a 'Filter Rows' input field, and 'Export' and 'Wrap Cell Contents' buttons. The result is labeled 'Result 21'.

no_order
400

2. Calculate the total revenue collected from all orders

```
select sum(total_amount) as Total_Revenue from orders;
```

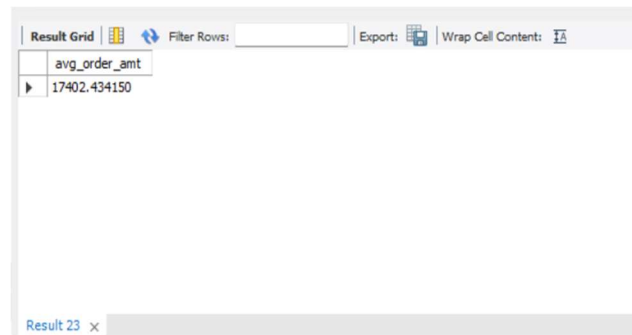


The screenshot shows a database query result grid. The grid has two columns: 'Total_Revenue' and a value '6960973.66'. The interface includes a 'Result Grid' tab, a 'Filter Rows' input field, and 'Export' and 'Wrap Cell Contents' buttons. The result is labeled 'Result 22'.

Total_Revenue
6960973.66

3. Calculate the average order value

```
select avg(total_amount) as avg_order_amt from orders;
```

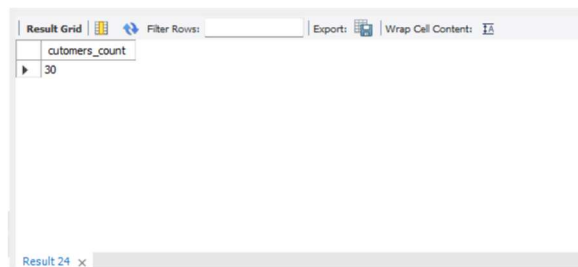


The screenshot shows a database query result grid. At the top, there are tabs for 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. Below the tabs, the column header is 'avg_order_amt'. The first row of data shows the value '17402.434150'. At the bottom left, there is a tab labeled 'Result 23'.

avg_order_amt
17402.434150

4. Count the number of customers who have placed at least one order

```
select count(distinct customer_id) as cutomers_count from orders;
```



The screenshot shows a database query result grid. At the top, there are tabs for 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. Below the tabs, the column header is 'cutomers_count'. The first row of data shows the value '30'. At the bottom left, there is a tab labeled 'Result 24'.

cutomers_count
30

5. Find the number of orders placed by each customer

`select customer_id, count(total_amount) as order_count from orders group by customer_id;`

Result Grid		Filter Rows:	Export:	Wrap Cell Content: IA
customer_id	order_count			
1	12			
2	17			
3	17			
4	9			
5	14			
6	16			
7	13			
8	10			
9	10			
10	13			
11	9			

Result 26 ×

6. Find total sales amount made by each customer

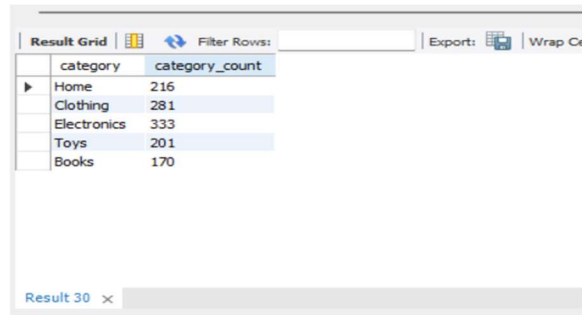
`select customer_id, sum(total_amount) as sales_amt from orders group by customer_id;`

Result Grid		Filter Rows:	Export:	Wrap Cell Content: IA
customer_id	sales_amt			
1	183747.44			
2	284420.07			
3	253783.31			
4	137562.22			
5	262504.19			
6	212173.58			
7	167960.11			
8	164701.78			
9	106226.03			
10	252722.75			
11	173409.01			

Result 28 ×

7. List the number of products sold per category

```
select p.category, count(p.category) as category_count  
from order_items oi  
left join products p  
on p.product_id=oi.product_id  
group by p.category;
```



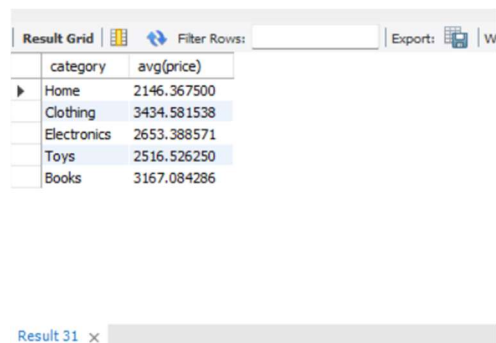
The screenshot shows a database query result grid with the following data:

category	category_count
Home	216
Clothing	281
Electronics	333
Toys	201
Books	170

Result 30

8. Find the average item price per category

```
select category, avg(price) from products  
group by category;
```



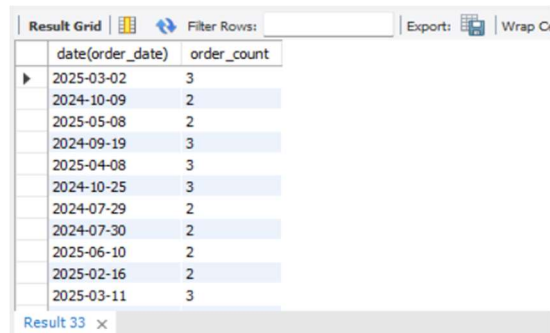
The screenshot shows a database query result grid with the following data:

category	avg(price)
Home	2146.367500
Clothing	3434.581538
Electronics	2653.388571
Toys	2516.526250
Books	3167.084286

Result 31

9. Show number of orders placed per day

```
select date(order_date),count(order_id) as order_count  
from orders  
group by date(order_date);
```



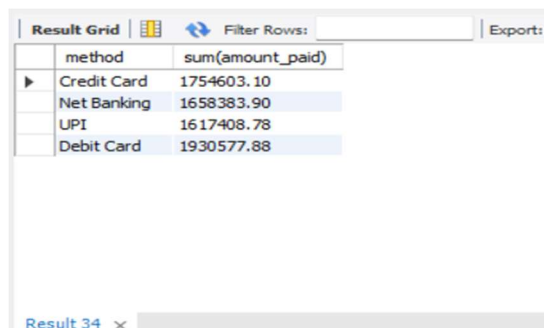
The screenshot shows a 'Result Grid' window with a table containing two columns: 'date(order_date)' and 'order_count'. The table lists 12 rows of data, showing dates and their corresponding order counts. The window also includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap C' option.

date(order_date)	order_count
2025-03-02	3
2024-10-09	2
2025-05-08	2
2024-09-19	3
2025-04-08	3
2024-10-25	3
2024-07-29	2
2024-07-30	2
2025-06-10	2
2025-02-16	2
2025-03-11	3

Result 33 x

10. List total payments received per payment method

```
select method,sum(amount_paid) from payments  
group by method;
```



The screenshot shows a 'Result Grid' window with a table containing two columns: 'method' and 'sum(amount_paid)'. The table lists four rows of data, showing payment methods and their corresponding total amounts. The window also includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap C' option.

method	sum(amount_paid)
Credit Card	1754603.10
Net Banking	1658383.90
UPI	1617408.78
Debit Card	1930577.88

Result 34 x

Level 4

1. Retrieve order details along with the customer name (INNER JOIN)

```
select o.*,c.name
from orders oi
inner join customers c
on o.customer_id= c.customer_id;
```

order_id	customer_id	order_date	status	total_amount	name
14	1	2024-09-24 21:21:38	Shipped	15803.34	Thomas Owens
17	1	2024-08-19 21:17:57	Pending	11173.77	Thomas Owens
61	1	2024-12-26 23:21:58	Shipped	13053.00	Thomas Owens
76	1	2025-01-14 22:59:54	Pending	23506.81	Thomas Owens
92	1	2024-09-26 11:33:58	Shipped	834.75	Thomas Owens
109	1	2025-03-17 04:56:18	Shipped	12190.14	Thomas Owens
127	1	2025-06-10 18:01:32	Pending	20160.64	Thomas Owens
135	1	2025-03-11 22:58:23	Pending	8891.79	Thomas Owens
144	1	2024-09-19 09:18:22	Shipped	12093.54	Thomas Owens
221	1	2024-09-25 05:49:39	Pending	12437.61	Thomas Owens
278	1	2025-02-01 20:05:38	Delivered	32015.16	Thomas Owens

2. Get list of products that have been sold (INNER JOIN with order_items)

```
select p.name as product_name,sum(oi.quantity) as total_quantity ,
sum(oi.item_price) as total_amt
from order_items oi
inner join products p
on p.product_id=oi.product_id
group by p.name;
```

product_name	total_quantity	total_amt
Plant No	43	14067.46
Population Social	33	77018.88
Available Answer	47	63237.75
Any Question	30	61870.64
Natural Network	44	99175.86
If Whatever	54	4789.80
Response Indeed	39	102844.56
Every Amount	61	112687.20
Common Study	57	25614.94
Development System	53	110440.94
Build Her	50	44463.36

3. List all orders with their payment method (INNER JOIN)

```
select o.*,p.method
from orders o
inner join payments p
on o.order_id=p.order_id;
```

order_id	customer_id	order_date	status	total_amount	method
1	20	2025-03-02 07:20:11	Delivered	9414.28	Credit Card
2	18	2024-10-09 18:08:21	Shipped	532.20	Net Banking
3	15	2025-05-08 00:08:27	Cancelled	5164.56	Credit Card
4	11	2024-09-19 22:16:13	Delivered	9469.78	UPI
5	12	2025-04-08 18:02:06	Pending	14501.86	UPI
6	29	2024-10-25 07:33:59	Cancelled	31050.17	UPI
7	22	2024-07-29 11:58:47	Shipped	3043.67	UPI
8	19	2024-07-30 22:49:49	Cancelled	32714.06	Net Banking
9	6	2025-06-10 17:00:25	Pending	24219.20	Net Banking
10	28	2025-02-16 12:45:59	Delivered	24342.52	Debit Card
11	25	2025-03-11 14:22:46	Cancelled	16196.13	Credit Card

4. Get list of customers and their orders (LEFT JOIN)

```
select c.customer_id,c.name, sum(total_amount)as amount_spent
from orders o
left join customers c
on c.customer_id= o.customer_id
group by c.customer_id
order by c.customer_id asc ;
```

customer_id	name	amount_spent
1	Thomas Owens	183747.44
2	Charles Grant	284420.07
3	Kaitlin Richards	253783.31
4	Christina Williams	137562.22
5	David Allen	262504.19
6	Mark Duke	212173.58
7	Briana Wright	167960.11
8	John Bryan	164701.78
9	Jason Thompson	106226.03
10	Shawn Hill	252722.75
11	Walter Jenkins	173409.01

5. List all products along with order item quantity (LEFT JOIN)

```
select p.name as product_name,sum(oi.quantity) as total_quantity
from order_items oi
left join products p
on p.product_id=oi.product_id
group by p.name
order by total_quantity desc;
```

Result Grid		Filter Rows:	Export:
product_name	total_quantity		
Assume Serve	83		
Age Treatment	67		
Some Them	63		
Southern Thing	62		
Every Amount	61		
Real Source	60		
Move Small	60		
Between Up	58		
Fire Often	58		
Everything Plant	58		
Group But	58		

Result 40 x

6. List all payments including those with no matching orders (RIGHT JOIN)

```
select p.*,o.total_amount as amt_as_per_orders
from orders o
right join payments p
on p.order_id=o.order_id;
```

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

	payment_id	order_id	payment_date	amount_paid	method	amt_as_per_orders
▶	1	1	2025-03-02 08:16:11	9414.28	Credit Card	9414.28
	2	2	2024-10-09 19:00:21	532.20	Net Banking	532.20
	3	3	2025-05-08 00:55:27	5164.56	Credit Card	5164.56
	4	4	2024-09-19 22:28:13	9469.78	UPI	9469.78
	5	5	2025-04-08 18:26:06	14501.86	UPI	14501.86
	6	6	2024-10-25 08:16:59	31050.17	UPI	31050.17
	7	7	2024-07-29 12:45:47	3043.67	UPI	3043.67
	8	8	2024-07-30 23:42:49	32714.06	Net Banking	32714.06
	9	9	2025-06-10 17:36:25	24219.20	Net Banking	24219.20
	10	10	2025-02-16 13:43:59	24342.52	Debit Card	24342.52
	11	11	2025-03-11 15:05:46	16196.13	Credit Card	16196.13

Result 41 x

7. Combine data from three tables: customer, order, and payment

```
select c.*, o.*, p.*
from orders o
left join customers c
on o.customer_id = c.customer_id
left join payments p
on o.order_id = p.order_id;
```

customer_id	name	email	phone	created_at	order_id	customer_id	order_date	status	total_amount	payment_id	order_id	payment_date	amount_paid	method
20	Megan Lee	user20@example.com	7044478333	2024-07-09 16:13:14	1	20	2025-03-02 07:20:11	Delivered	9414.28	1	1	2025-03-02 08:16:11	9414.28	Credit Card
18	Jeffrey Bray	user18@example.com	164.962.0222x92	2024-07-02 19:51:55	2	18	2024-10-09 18:08:21	Shipped	532.20	2	2	2024-10-09 19:00:21	532.20	Net Banking
15	Austin Flores	user15@example.com	329.901.1576x66	2024-06-13 09:03:42	3	15	2025-05-08 00:08:27	Cancelled	5164.56	3	3	2025-05-08 00:55:27	5164.56	Credit Card
11	Walter Jenkins	user11@example.com	536-329-0817x71	2023-10-26 03:12:30	4	11	2024-09-19 22:16:13	Delivered	9469.78	4	4	2024-09-19 22:28:13	9469.78	UPI
12	Mary Knight	user12@example.com	361-636-3802	2023-08-16 20:05:50	5	12	2025-04-08 18:02:06	Pending	14501.86	5	5	2025-04-08 18:26:06	14501.86	UPI
29	Kara Zavala	user29@example.com	434-985-5776x51	2023-09-27 15:57:09	6	29	2024-10-25 07:33:59	Cancelled	31050.17	6	6	2024-10-25 08:16:59	31050.17	UPI
22	Peter Phillips	user22@example.com	389-656-1695	2025-05-22 06:12:06	7	22	2024-07-29 11:58:47	Shipped	3043.67	7	7	2024-07-29 12:45:47	3043.67	UPI
19	Amanda Bright	user19@example.com	380.981.9798x69	2024-12-20 22:58:15	8	19	2024-07-30 22:49:49	Cancelled	32714.06	8	8	2024-07-30 23:42:49	32714.06	Net Banking
6	Mark Duke	user6@example.com	(144)957-2811	2024-06-24 03:22:59	9	6	2025-06-10 17:00:25	Pending	24219.20	9	9	2025-06-10 17:36:25	24219.20	Net Banking
28	Joseph Stuart	user28@example.com	363-045-4287	2023-09-14 05:52:25	10	28	2025-02-16 12:45:59	Delivered	24342.52	10	10	2025-02-16 13:43:59	24342.52	Debit Card

Result 42 x

Level 5

1. List all products priced above the average product price

```
select * from products where price > (select avg(price) from products);
```

product_id	name	category	price	stock_quantity	added_on
2	Population Social	Clothing	4813.68	84	2025-05-30 10:02:50
4	Any Question	Clothing	4759.28	179	2025-06-03 13:34:03
5	Natural Network	Toys	4722.66	75	2023-11-06 00:47:37
7	Response Indeed	Clothing	4897.36	36	2025-03-29 02:43:08
8	Every Amount	Home	4173.60	156	2025-04-30 03:11:10
10	Development System	Electronics	4801.78	153	2025-03-12 08:22:57
12	Action Ask	Electronics	4017.01	19	2025-02-14 03:38:06
18	Some Them	Toys	3673.86	110	2024-07-25 00:33:53
19	Build High	Clothing	4707.14	47	2023-09-01 02:50:01
20	Real Source	Books	4398.66	197	2025-02-08 10:28:27
21	Bed Including	Clothing	3845.31	92	2025-01-21 11:52:40

products 43 x

2. Find customers who have placed at least one order

`select distinct customer_id from orders;`

The screenshot shows a 'Result Grid' window with a 'Filter Rows' field and an 'Export' button. The table has one column, 'customer_id', and 11 rows of data. The values are 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, and 11. A status bar at the bottom indicates 'orders 44'.

customer_id
1
2
3
4
5
6
7
8
9
10
11

3. Show orders whose total amount is above the average for that customer

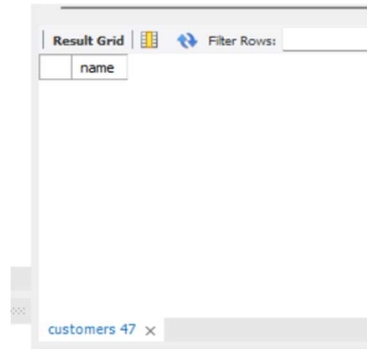
`select * from orders o
where total_amount >
(select avg(total_amount) from orders
where customer_id=o.customer_id);`

The screenshot shows a 'Result Grid' window with a 'Filter Rows' field, 'Edit', 'Export/Import', and 'Print' buttons. The table has five columns: 'order_id', 'customer_id', 'order_date', 'status', and 'total_amount'. It displays 14 rows of data. A status bar at the bottom indicates 'orders 46'.

order_id	customer_id	order_date	status	total_amount
6	29	2024-10-25 07:33:59	Cancelled	31050.17
8	19	2024-07-30 22:49:49	Cancelled	32714.06
9	6	2025-06-10 17:00:25	Pending	24219.20
10	28	2025-02-16 12:45:59	Delivered	24342.52
14	1	2024-09-24 21:21:38	Shipped	15803.34
16	5	2024-09-09 03:00:40	Shipped	22856.73
18	13	2024-06-15 12:57:04	Cancelled	32001.24
21	22	2025-03-12 18:53:46	Pending	25364.11
22	20	2025-02-21 11:21:06	Shipped	21281.44
24	6	2024-06-19 07:09:04	Pending	22882.19
25	17	2024-07-24 12:56:24	Pending	23749.66

4. Display customers who haven't placed any orders

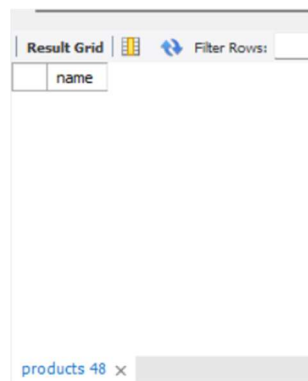
```
select name from customers where customer_id not in  
(select customer_id from orders);
```



The screenshot shows a database query result grid. At the top, there is a tab labeled 'Result Grid' and a 'Filter Rows' section. Below the tab, a single column header 'name' is visible. The grid itself is empty. At the bottom of the window, a status bar indicates 'customers 47' with a close button (X).

5. Show products that were never ordered

```
select name from products where product_id not in  
(select product_id from orders);
```



The screenshot shows a database query result grid. At the top, there is a tab labeled 'Result Grid' and a 'Filter Rows' section. Below the tab, a single column header 'name' is visible. The grid itself is empty. At the bottom of the window, a status bar indicates 'products 48' with a close button (X).

6. Show highest value order per customer

```
select customer_id, max(total_amount) as highest_value from orders
group by customer_id;
```

Result Grid		Filter Rows:	Export
customer_id	highest_value		
1	32015.16		
2	42056.04		
3	41679.11		
4	25747.34		
5	39921.78		
6	39003.19		
7	28589.04		
8	36147.09		
9	21414.44		
10	35723.17		
11	37129.82		

Result 50 x

7. Highest Order Per Customer (Including Names)

```
select c.customer_id,c.name,max(o.total_amount) as Highest_order_amt
from orders o
left join customers c
on c.customer_id=o.customer_id
group by customer_id;
```

Result Grid		Filter Rows:	Export	Wrap C
customer_id	name	Highest_order_amt		
20	Megan Lee	39563.51		
18	Jeffrey Bray	39857.19		
15	Austin Flores	40510.54		
11	Walter Jenkins	37129.82		
12	Mary Knight	38656.26		
29	Kara Zavala	43867.08		
22	Peter Phillips	36504.68		
19	Amanda Bright	35676.00		
6	Mark Duke	39003.19		
28	Joseph Stuart	44504.56		
25	Cindy Hart	40872.29		

Result 51 x

Level 6

1. List all customers who have either placed an order or written a product review

```
select customer_id from orders
```

Union

```
select customer_id from product_reviews;
```

Result Grid | Filter Rows:

customer_id
1
2
3
4
5
6
7
8
9
10
11

Result 52 x

2. List all customers who have placed an order as well as reviewed a product

```
select distinct customer_id from orders where customer_id in  
(select customer_id from product_reviews);
```

Result Grid | Filter Rows:

customer_id
1
2
4
6
7
9
10
11
13
14
15

orders 54 x