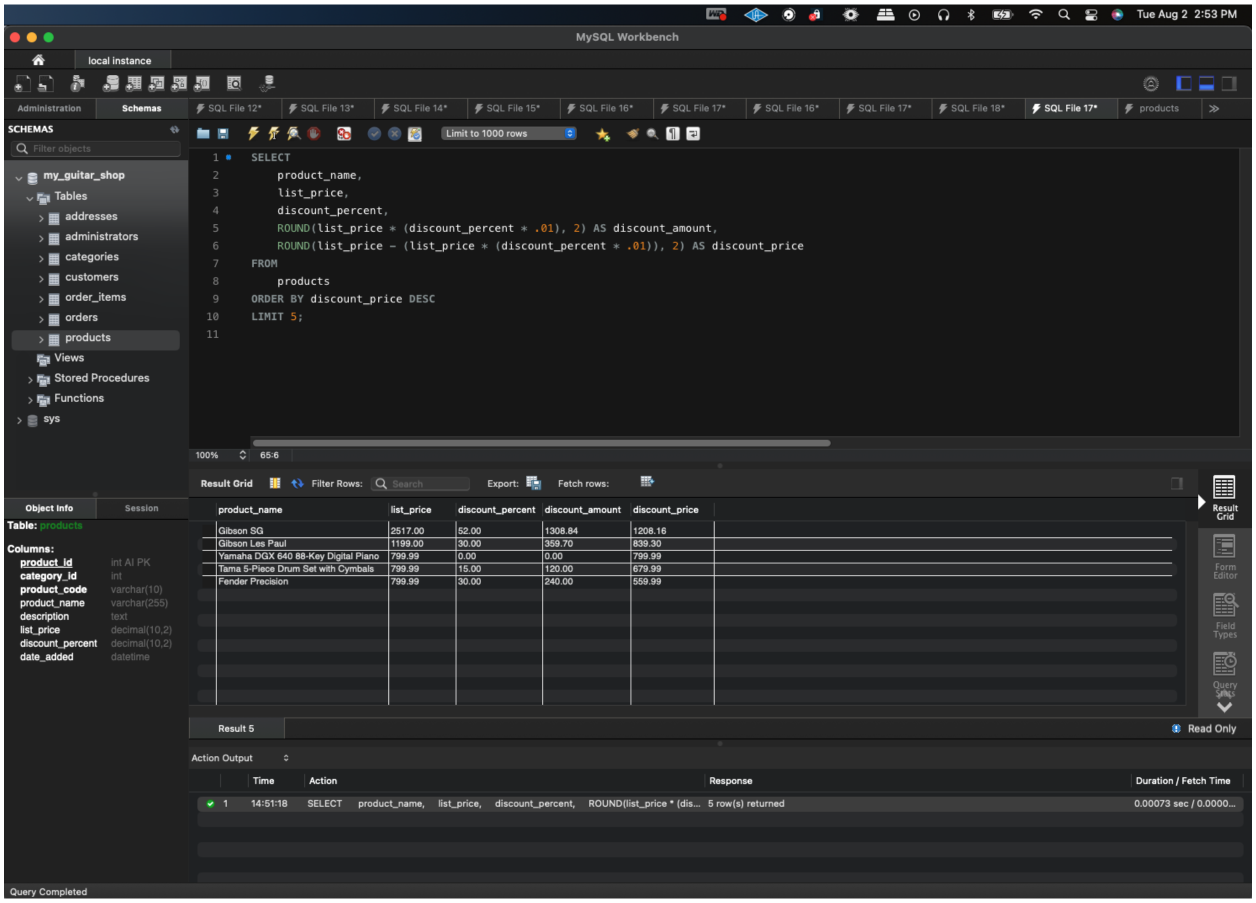
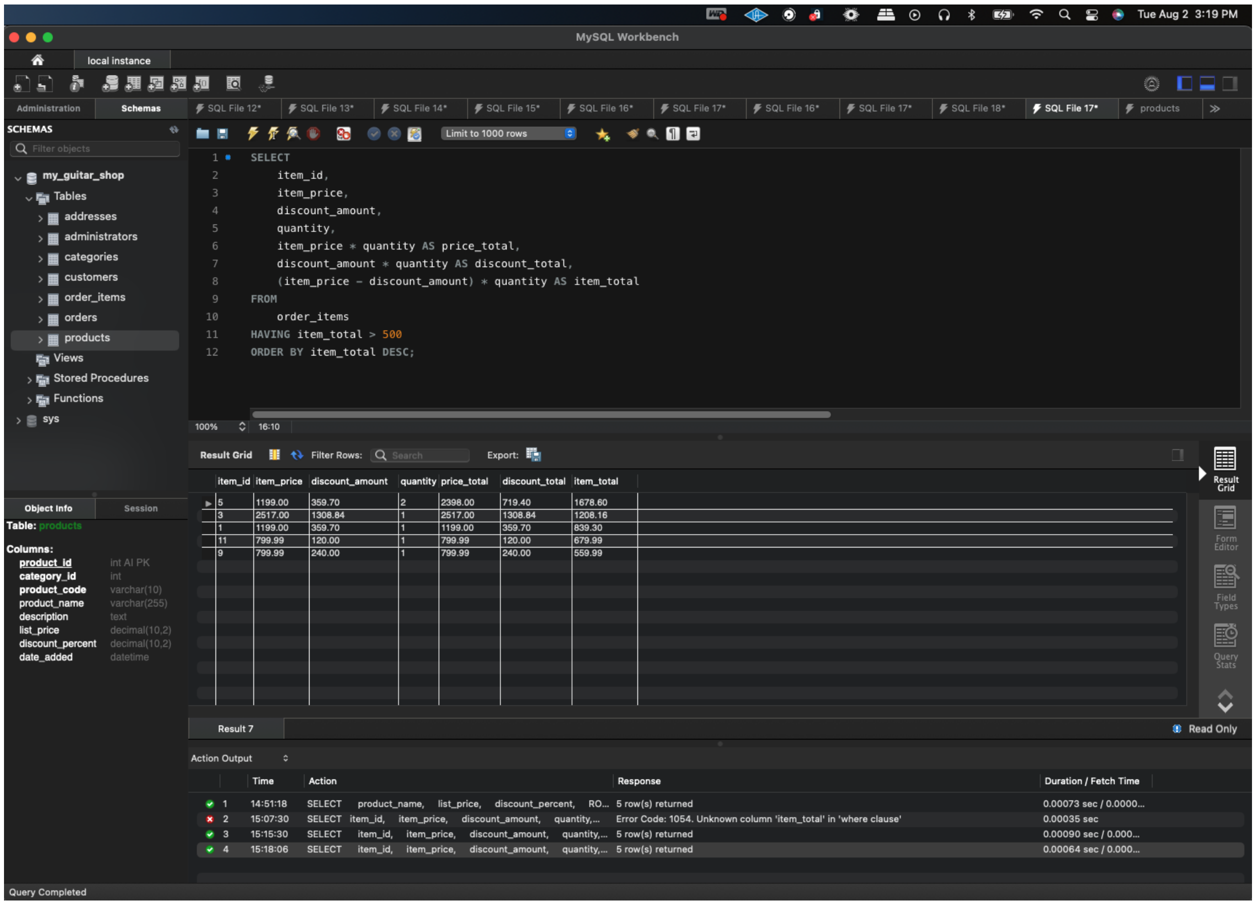


```sql
SELECT
    product_name,
    list_price,
    discount_percent,
    ROUND(list_price * (discount_percent * .01), 2) AS discount_amount,
    ROUND(list_price - (list_price * (discount_percent * .01)), 2) AS discount_price
FROM
    products
ORDER BY discount_price DESC
LIMIT 5;
```

product_name	list_price	discount_percent	discount_amount	discount_price
Gibson SG	2517.00	52.00	1308.84	1208.16
Gibson Les Paul	1199.00	30.00	359.70	839.30
Yamaha DGX 640 88-Key Digital Piano	799.99	0.00	0.00	799.99
Tama 5-Piece Drum Set with Cymbals	799.99	15.00	120.00	679.99
Fender Precision	799.99	30.00	240.00	559.99

```
1   14:51:18   SELECT   product_name,   list_price,   discount_percent,   ROUND(list_price * (dis...   5 row(s) returned   0.00073 sec / 0.0000...
```

This query which selects the product name, list price and discount percent columns from the products table. The query also creates a column called discount_amount which is calculated from the discount percent and list price columns. The query creates another column called discount_price which takes the same calculation from the discount_amount column but subtracts it with the list_price value. Both of these created columns are rounded to the nearest hundredth. The query is in descending order by the discount price column and is limited to return the first five rows.

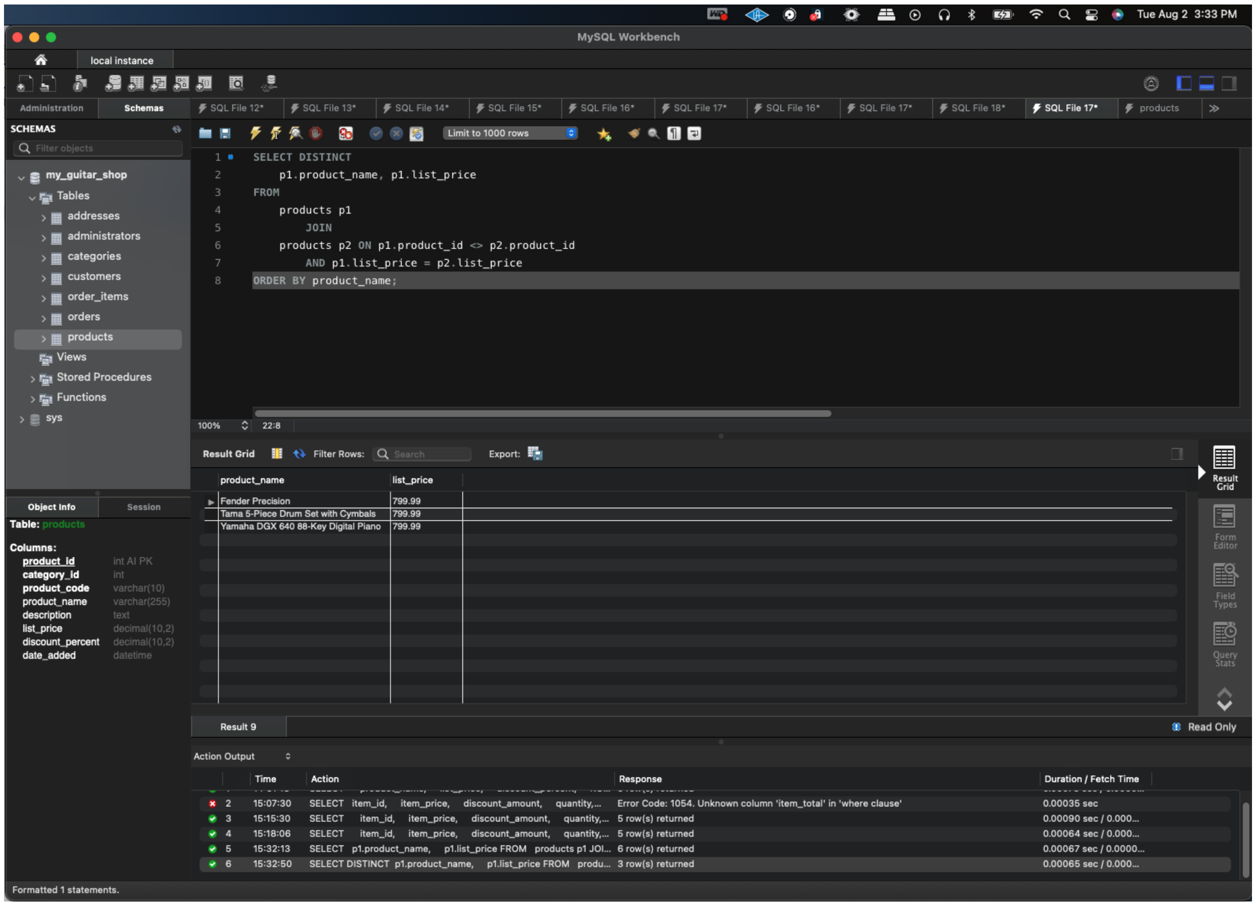


```
SELECT
    item_id,
    item_price,
    discount_amount,
    quantity,
    item_price * quantity AS price_total,
    discount_amount * quantity AS discount_total,
    (item_price - discount_amount) * quantity AS item_total
FROM
    order_items
HAVING item_total > 500
ORDER BY item_total DESC;
```

item_id	item_price	discount_amount	quantity	price_total	discount_total	item_total
5	1199.00	359.70	2	2398.00	719.40	1678.60
3	2517.00	1308.84	1	2517.00	1308.84	1208.16
1	1199.00	359.70	1	1199.00	359.70	839.30
11	799.99	120.00	1	799.99	120.00	679.99
9	799.99	240.00	1	799.99	240.00	559.99

```
   Time      Action                                                     Response                                                        Duration / Fetch Time
1  14:51:18  SELECT  product_name,  list_price,  discount_percent,  RO...  5 row(s) returned                                              0.00073 sec / 0.0000...
2  15:07:30  SELECT  item_id,  item_price,  discount_amount,  quantity,...  Error Code: 1054. Unknown column 'item_total' in 'where clause'  0.00035 sec
3  15:15:30  SELECT  item_id,  item_price,  discount_amount,  quantity,...  5 row(s) returned                                              0.00090 sec / 0.000...
4  15:18:06  SELECT  item_id,  item_price,  discount_amount,  quantity,...  5 row(s) returned                                              0.00064 sec / 0.000...
```

This query which selects the item id, item price, discount amount and quantity columns from the order_items table. The query also creates three columns called price_total, discount_total and item_total. The price_total is calculated by the product of the item price and quantity. The discount total is calculated from the product of theitem price and quantity. The item total is calculated from the subtraction of the item price and discount amount all multiplied by the quantity. Lastly, the query is in descending order by the item total and only displays results which are above the item total of 500.

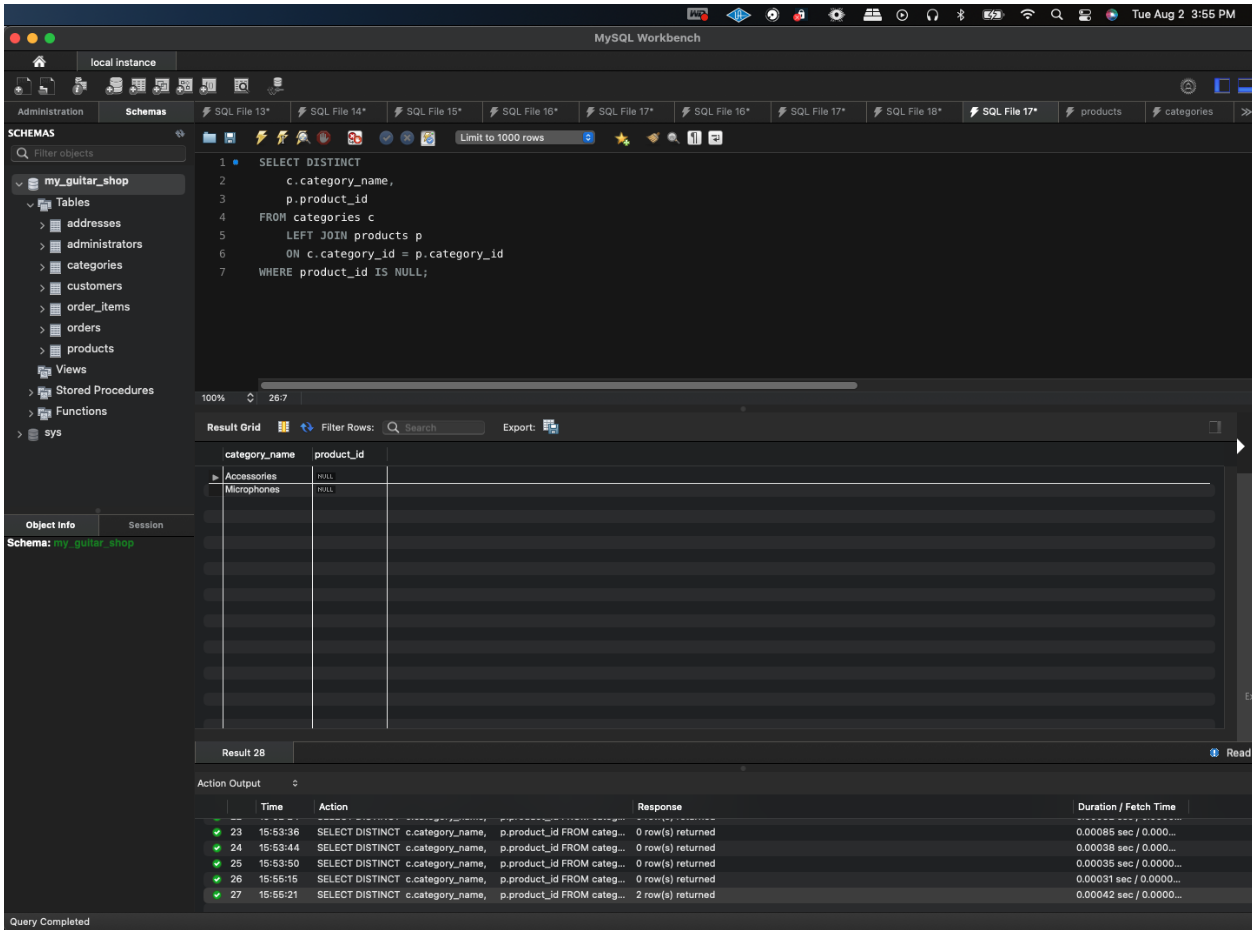


```
SELECT DISTINCT
    p1.product_name, p1.list_price
FROM
    products p1
        JOIN
    products p2 ON p1.product_id <> p2.product_id
        AND p1.list_price = p2.list_price
ORDER BY product_name;
```

product_name	list_price
Fender Precision	799.99
Tama 5-Piece Drum Set with Cymbals	799.99
Yamaha DGX 640 88-Key Digital Piano	799.99

```
Action Output
    Time      Action                                                      Response                                                       Duration / Fetch Time
  2 15:07:30  SELECT  item_id,  item_price,  discount_amount,  quantity,... Error Code: 1054. Unknown column 'item_total' in 'where clause'   0.00035 sec
  3 15:15:30  SELECT  item_id,  item_price,  discount_amount,  quantity,... 5 row(s) returned                                              0.00090 sec / 0.000...
  4 15:18:06  SELECT  item_id,  item_price,  discount_amount,  quantity,... 5 row(s) returned                                              0.00064 sec / 0.000...
  5 15:32:13  SELECT  p1.product_name,  p1.list_price FROM  products p1 JOI... 6 row(s) returned                                           0.00067 sec / 0.0000...
  6 15:32:50  SELECT DISTINCT  p1.product_name,  p1.list_price FROM  produ... 3 row(s) returned                                            0.00065 sec / 0.000...
```

This query which uses a self join to return the distinct rows of the product name and the list price columns. The query uses aliasing and comparison operators to complete the self join and is ordered by the product name column.

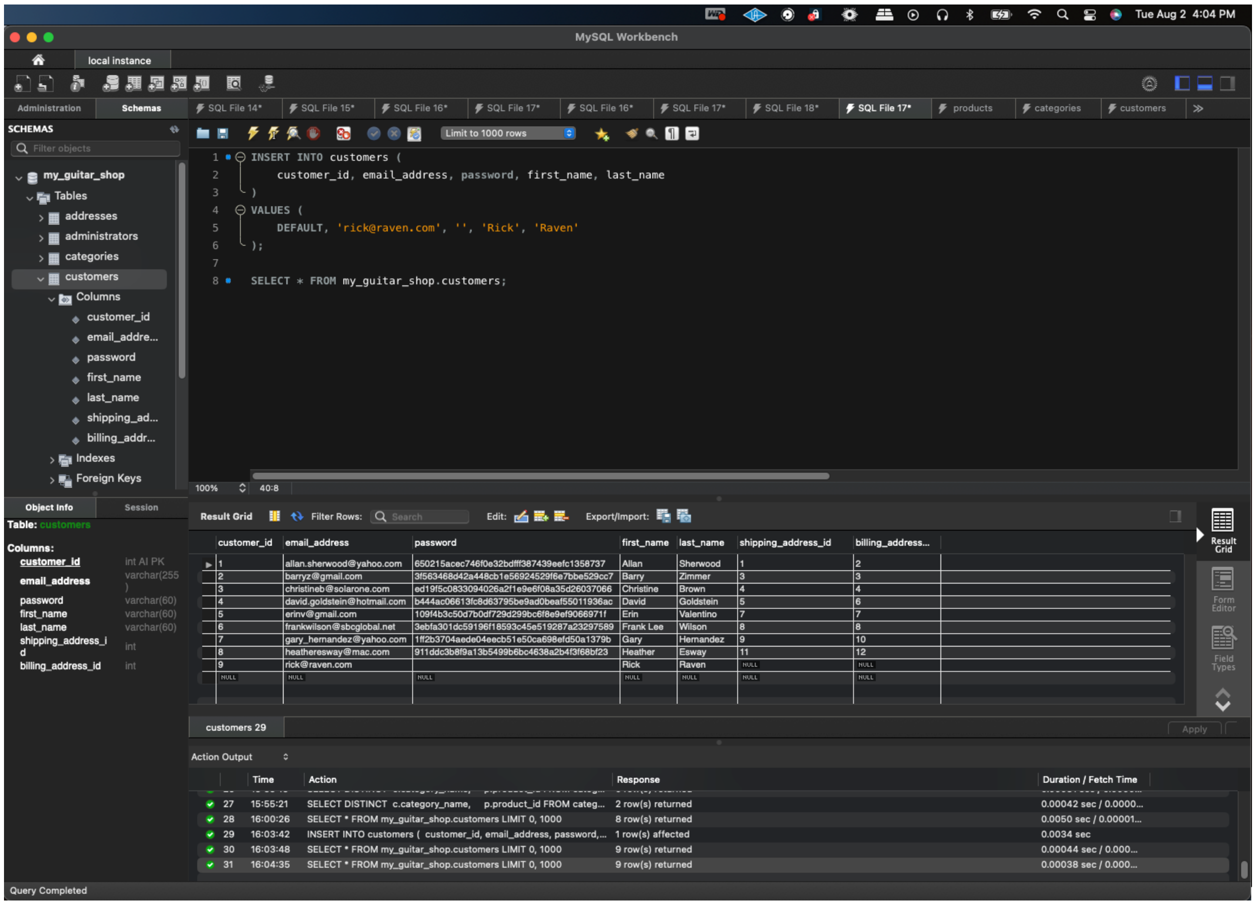


```
SELECT DISTINCT
    c.category_name,
    p.product_id
FROM categories c
    LEFT JOIN products p
    ON c.category_id = p.category_id
WHERE product_id IS NULL;
```

category_name	product_id
Accessories	NULL
Microphones	NULL

```
Action Output
    Time      Action                                                         Response          Duration / Fetch Time
23  15:53:36  SELECT DISTINCT  c.category_name,  p.product_id FROM categ...  0 row(s) returned  0.00085 sec / 0.000...
24  15:53:44  SELECT DISTINCT  c.category_name,  p.product_id FROM categ...  0 row(s) returned  0.00038 sec / 0.000...
25  15:53:50  SELECT DISTINCT  c.category_name,  p.product_id FROM categ...  0 row(s) returned  0.00035 sec / 0.0000...
26  15:55:15  SELECT DISTINCT  c.category_name,  p.product_id FROM categ...  0 row(s) returned  0.00031 sec / 0.0000...
27  15:55:21  SELECT DISTINCT  c.category_name,  p.product_id FROM categ...  2 row(s) returned  0.00042 sec / 0.0000...
```

This query which selects each row of the category name and the product id column which has not been used. This is completed by an outer join on the category id column with the search condition of product id being a null value.



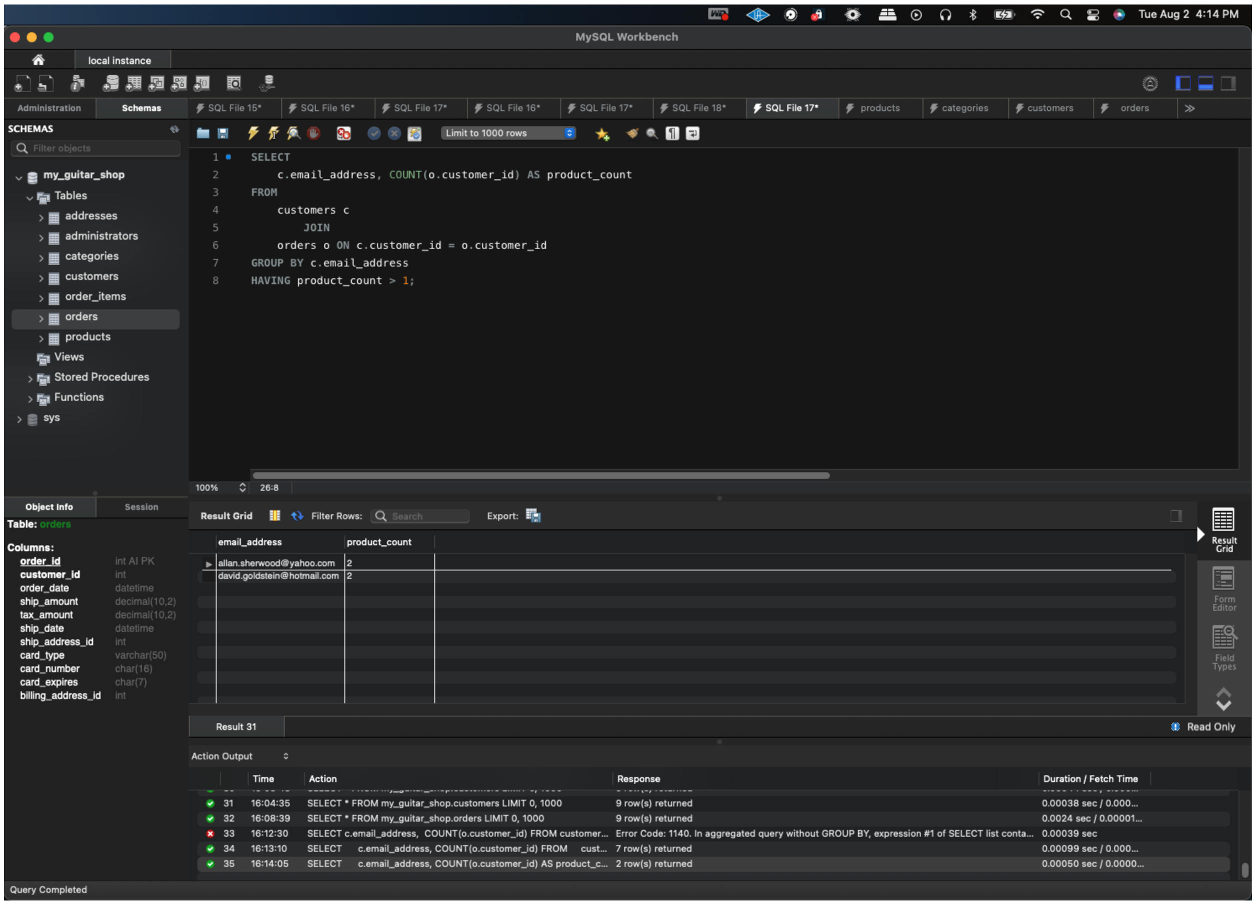
```sql
INSERT INTO customers (
    customer_id, email_address, password, first_name, last_name
)
VALUES (
    DEFAULT, 'rick@raven.com', '', 'Rick', 'Raven'
);

SELECT * FROM my_guitar_shop.customers;
```

customer_id	email_address	password	first_name	last_name	shipping_address_id	billing_address...
1	allan.sherwood@yahoo.com	650215acec746f0e32bdfff387439eefc1358737	Allan	Sherwood	1	2
2	barryz@gmail.com	3f563468d42a448cb1e56924529f6e7bbe529cc7	Barry	Zimmer	3	3
3	christineb@solarone.com	ed19f5c0833094026a2f1e9e6f08a35d26037066	Christine	Brown	4	4
4	david.goldstein@hotmail.com	b444ac06613fc8d63795be9ad0beaf55011936ac	David	Goldstein	5	6
5	erinv@gmail.com	109f4b3c50d7b0df729d299bc6f8e9ef9066971f	Erin	Valentino	7	7
6	frankwilson@sbcglobal.net	3ebfa301dc59196f18593c45e519287a23297589	Frank Lee	Wilson	8	8
7	gary_hernandez@yahoo.com	1ff2b3704aede04eecb51e50ca698efd50a1379b	Gary	Hernandez	9	10
8	heatheresway@mac.com	911ddc3b8f9a13b5499b6bc4638a2b4f3f68bf23	Heather	Esway	11	12
9	rick@raven.com		Rick	Raven	NULL	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
     Time      Action                                                    Response            Duration / Fetch Time
27   15:55:21  SELECT DISTINCT  c.category_name,    p.product_id FROM categ...  2 row(s) returned   0.00042 sec / 0.0000...
28   16:00:26  SELECT * FROM my_guitar_shop.customers LIMIT 0, 1000       8 row(s) returned   0.0050 sec / 0.00001...
29   16:03:42  INSERT INTO customers (  customer_id, email_address, password,...  1 row(s) affected   0.0034 sec
30   16:03:48  SELECT * FROM my_guitar_shop.customers LIMIT 0, 1000       9 row(s) returned   0.00044 sec / 0.000...
31   16:04:35  SELECT * FROM my_guitar_shop.customers LIMIT 0, 1000       9 row(s) returned   0.00038 sec / 0.000...
```

This query which inserts a row into the customers table. This insert operator defaults the customer id, has specific values for the next four columns and leaves the shipping address id and billing address id values null.



```sql
SELECT
    c.email_address, COUNT(o.customer_id) AS product_count
FROM
    customers c
        JOIN
    orders o ON c.customer_id = o.customer_id
GROUP BY c.email_address
HAVING product_count > 1;
```

email_address	product_count
allan.sherwood@yahoo.com	2
david.goldstein@hotmail.com	2

Table: orders

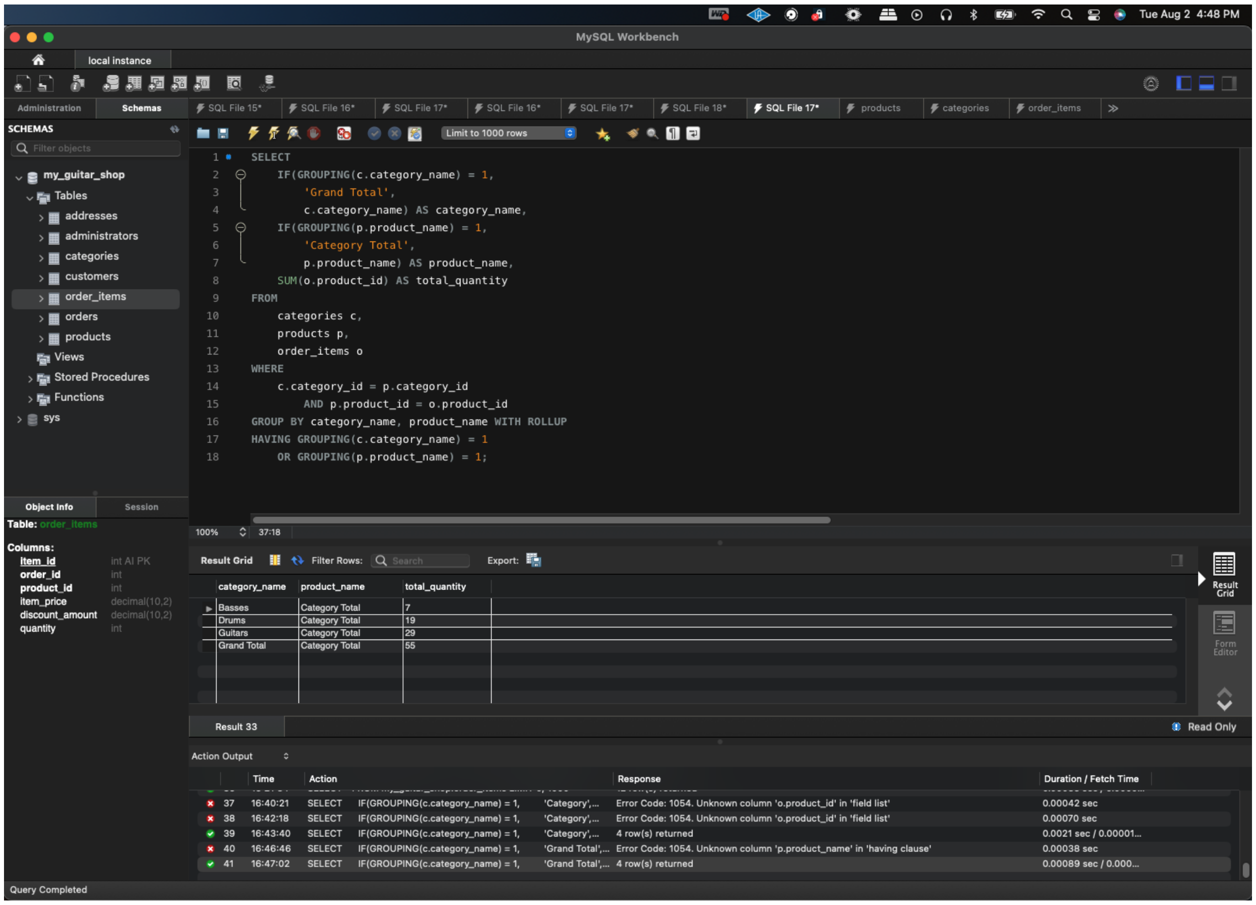
Columns:

order_id	int AI PK
customer_id	int
order_date	datetime
ship_amount	decimal(10,2)
tax_amount	decimal(10,2)
ship_date	datetime
ship_address_id	int
card_type	varchar(50)
card_number	char(16)
card_expires	char(7)
billing_address_id	int

```
    Time      Action                                                        Response                                                                          Duration / Fetch Time
31  16:04:35  SELECT * FROM my_guitar_shop.customers LIMIT 0, 1000          9 row(s) returned                                                                 0.00038 sec / 0.000...
32  16:08:39  SELECT * FROM my_guitar_shop.orders LIMIT 0, 1000             9 row(s) returned                                                                 0.0024 sec / 0.00001...
33  16:12:30  SELECT c.email_address,  COUNT(o.customer_id) FROM customer...  Error Code: 1140. In aggregated query without GROUP BY, expression #1 of SELECT list conta...  0.00039 sec
34  16:13:10  SELECT     c.email_address, COUNT(o.customer_id) FROM    cust...  7 row(s) returned                                                                 0.00099 sec / 0.000...
35  16:14:05  SELECT     c.email_address, COUNT(o.customer_id) AS product_c...  2 row(s) returned                                                                 0.00050 sec / 0.0000...
```

Query Completed

This query which selects the email address column and the count of the customer id as a column named product_count. This query joins the tables and sorts them by the email address and only displays the rows of the customers who have more than one product purchase.

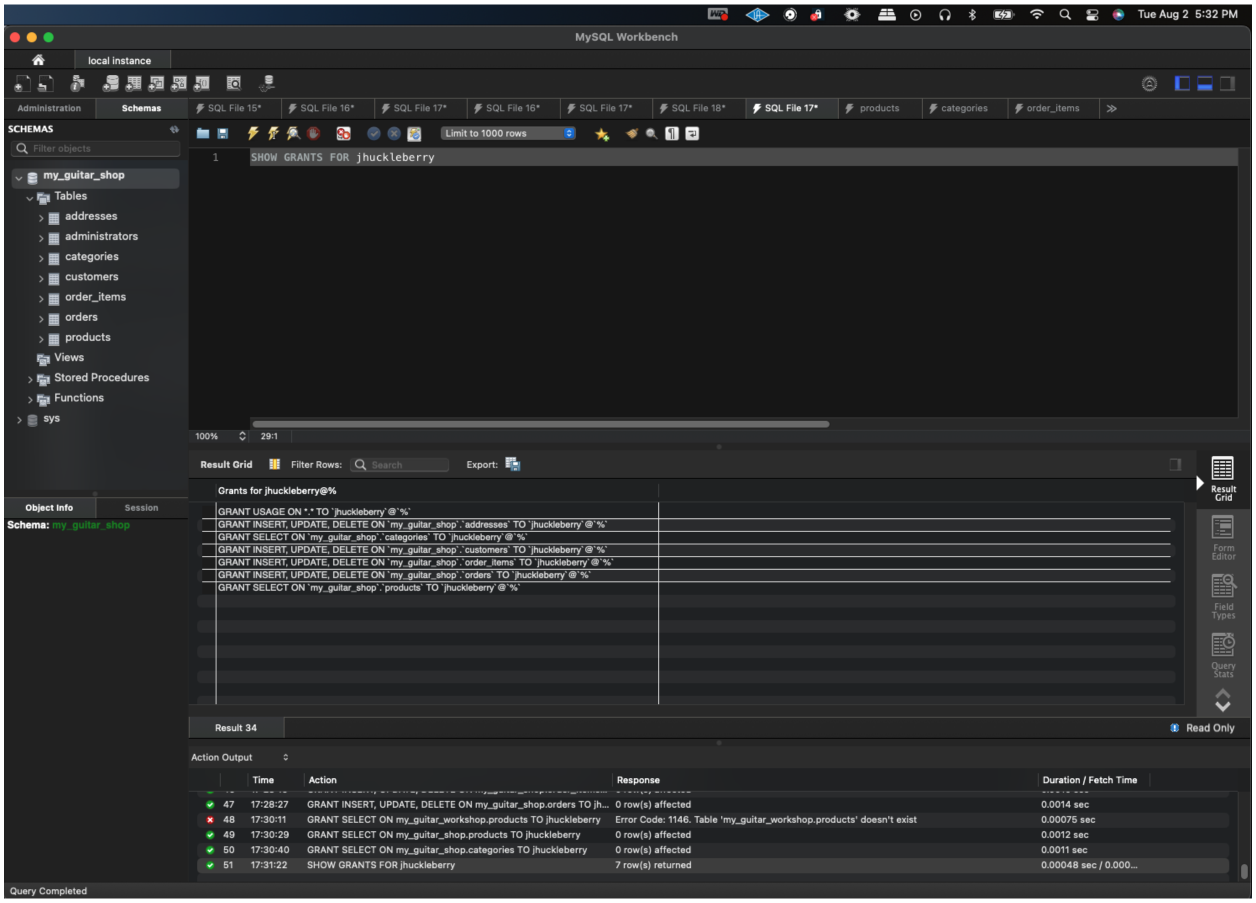


```sql
SELECT
    IF(GROUPING(c.category_name) = 1,
        'Grand Total',
        c.category_name) AS category_name,
    IF(GROUPING(p.product_name) = 1,
        'Category Total',
        p.product_name) AS product_name,
    SUM(o.product_id) AS total_quantity
FROM
    categories c,
    products p,
    order_items o
WHERE
    c.category_id = p.category_id
        AND p.product_id = o.product_id
GROUP BY category_name, product_name WITH ROLLUP
HAVING GROUPING(c.category_name) = 1
    OR GROUPING(p.product_name) = 1;
```

category_name	product_name	total_quantity
Basses	Category Total	7
Drums	Category Total	19
Guitars	Category Total	29
Grand Total	Category Total	55

```
    Time      Action                                                        Response                                                          Duration / Fetch Time
37  16:40:21  SELECT  IF(GROUPING(c.category_name) = 1,   'Category',...    Error Code: 1054. Unknown column 'o.product_id' in 'field list'   0.00042 sec
38  16:42:18  SELECT  IF(GROUPING(c.category_name) = 1,   'Category',...    Error Code: 1054. Unknown column 'o.product_id' in 'field list'   0.00070 sec
39  16:43:40  SELECT  IF(GROUPING(c.category_name) = 1,   'Category',...    4 row(s) returned                                                 0.0021 sec / 0.00001...
40  16:46:46  SELECT  IF(GROUPING(c.category_name) = 1,   'Grand Total',... Error Code: 1054. Unknown column 'p.product_name' in 'having clause'  0.00038 sec
41  16:47:02  SELECT  IF(GROUPING(c.category_name) = 1,   'Grand Total',... 4 row(s) returned                                                 0.00089 sec / 0.000...
```

This query which selects the category name, product name and creates a column called total_quantity which is the aggregate sum of the products from each category. Essentially this query returns the total quantity of products which were purchased from each category. This also includes a grand total row which is created by the WITH ROLLUP operator to create a summary of the data and replaces the null values that would be in that row with literals.



```
SHOW GRANTS FOR jhuckleberry
```

Grants for jhuckleberry@%
GRANT USAGE ON *.* TO \`jhuckleberry\`@\`%\`
GRANT INSERT, UPDATE, DELETE ON \`my_guitar_shop\`.\`addresses\` TO \`jhuckleberry\`@\`%\`
GRANT SELECT ON \`my_guitar_shop\`.\`categories\` TO \`jhuckleberry\`@\`%\`
GRANT INSERT, UPDATE, DELETE ON \`my_guitar_shop\`.\`customers\` TO \`jhuckleberry\`@\`%\`
GRANT INSERT, UPDATE, DELETE ON \`my_guitar_shop\`.\`order_items\` TO \`jhuckleberry\`@\`%\`
GRANT INSERT, UPDATE, DELETE ON \`my_guitar_shop\`.\`orders\` TO \`jhuckleberry\`@\`%\`
GRANT SELECT ON \`my_guitar_shop\`.\`products\` TO \`jhuckleberry\`@\`%\`

Action Output

	Time	Action	Response	Duration / Fetch Time
47	17:28:27	GRANT INSERT, UPDATE, DELETE ON my_guitar_shop.orders TO jh...	0 row(s) affected	0.0014 sec
48	17:30:11	GRANT SELECT ON my_guitar_workshop.products TO jhuckleberry	Error Code: 1146. Table 'my_guitar_workshop.products' doesn't exist	0.00075 sec
49	17:30:29	GRANT SELECT ON my_guitar_shop.products TO jhuckleberry	0 row(s) affected	0.0012 sec
50	17:30:40	GRANT SELECT ON my_guitar_shop.categories TO jhuckleberry	0 row(s) affected	0.0011 sec
51	17:31:22	SHOW GRANTS FOR jhuckleberry	7 row(s) returned	0.00048 sec / 0.000...

Query Completed

This query which shows the granted privileges to a user which was created with my first initial and last name. This user has a password and can access MySQL from any computer. The user has insert, update and delete privileges for the addresses, customers, orders and order items tables of the my guitar shop database. This user also has privileges to use the select operator on the categories and products tables within the my guitar shop database.