

Stored Procedures:

1)

NAME: GetCustomerFeedbackWith

This procedure takes a customer ID as an input parameter and outputs CustomerFirstName, CustomerLastName, CustomerPhoneNo, CustomerEmail, orderID, productRating, shippingRating, ExperienceRating form Customer, CustomerFeedback table.

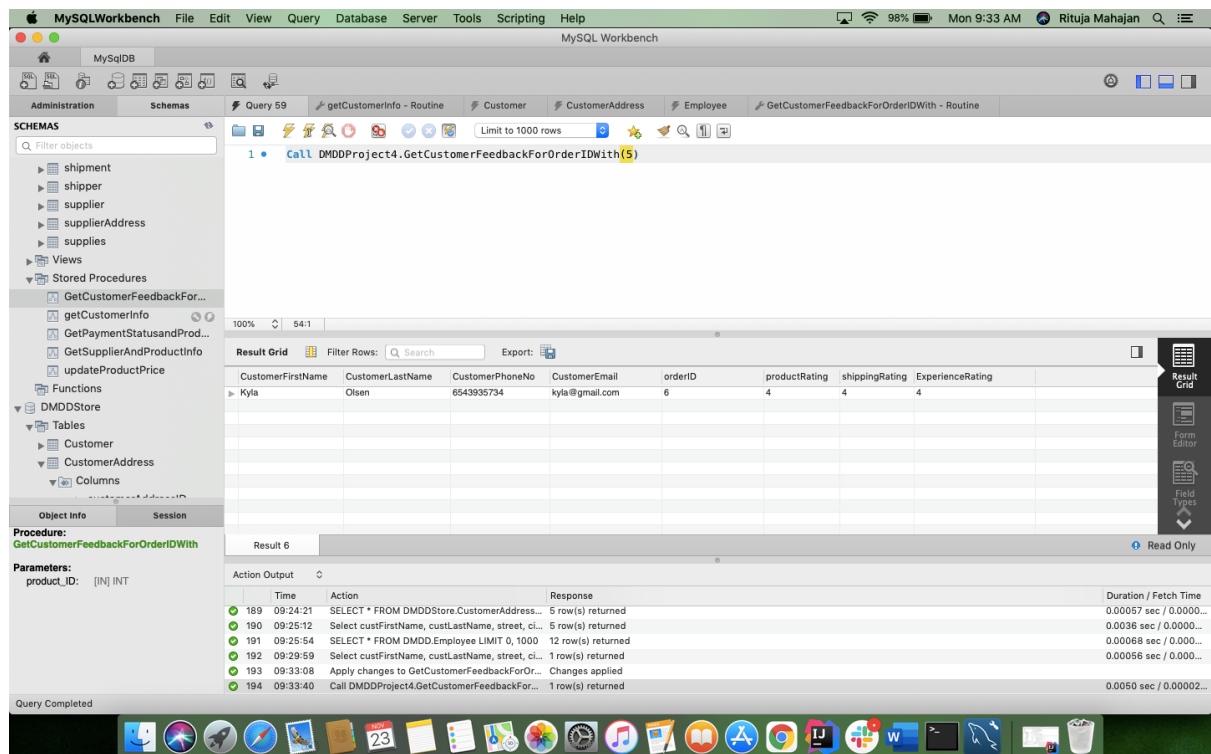
CODE:

```
CREATE PROCEDURE `GetCustomerFeedbackWith` (IN customer_ID INT)
BEGIN
    SELECT CustomerFirstName, CustomerLastName, CustomerPhoneNo,
    CustomerEmail, orderID, productRating, shippingRating, ExperienceRating
    FROM DMDDP4.Customer, DMDDP4.customerFeedback
    WHERE customerFeedback.customerID = customer_ID;
END
```

EXECUTION:

Call DMDDP4. GetCustFeedbackForOrderIDWithCustID(5);

OUTPUT:



The screenshot shows the MySQL Workbench interface. The 'Schemas' pane on the left lists the database structure, including tables and stored procedures. The 'Query' pane shows the execution of the stored procedure `DMDDProject4.GetCustomerFeedbackForOrderIDWith(5)`. The 'Result Grid' displays the output of the procedure, which is a single row of customer data. The 'Action Output' pane shows the sequence of SQL statements executed during the procedure call.

CustomerFirstName	CustomerLastName	CustomerPhoneNo	CustomerEmail	orderID	productRating	shippingRating	ExperienceRating
Kyla	Olsen	6543935734	kyla@gmail.com	6	4	4	4

Procedure: GetCustomerFeedbackForOrderIDWith
Parameters: product_ID: [IN] INT

Action Output:

Time	Action	Response	Duration / Fetch Time
189 09:24:21	SELECT * FROM DMDDStore.CustomerAddress...	5 row(s) returned	0.00057 sec / 0.0000...
190 09:25:12	Select custFirstName, custLastName, street, ci...	5 row(s) returned	0.0036 sec / 0.0000...
191 09:25:54	SELECT * FROM DMDD.Employee LIMIT 0, 1000	12 row(s) returned	0.00068 sec / 0.000...
192 09:29:59	Select custFirstName, custLastName, street, ci...	1 row(s) returned	0.00056 sec / 0.000...
193 09:33:08	Apply changes to GetCustomerFeedbackForOr...	Changes applied	
194 09:33:40	Call DMDDProject4.GetCustomerFeedbackFor...	1 row(s) returned	0.0050 sec / 0.00002...

2)

NAME: GetCustomerInformation

This procedure takes a customer ID as an input parameter and outputs Customer FirstName, Customer Last Name, Customer Phone No, Customer Email, street, city, postal Code form Customer, Customer Address table.

CODE:

```
CREATE PROCEDURE `GetCustomerInformation`(IN customer_ID INT)
BEGIN
    Select CustomerFirstName, CustomerLastName, CustomerPhoneNO,
    CustomerEmail, street, city, postalCode
    FROM DMDD4.Customer, DMDD4.customerAddress
    WHERE Customer.customerID = customer_ID and customerAddress.customerID =
    customer_ID;
END
```

EXECUTION:

CALL DMDDP4. GetCustomerInformation(5)

OUTPUT

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane lists various databases and tables. The central 'Query Editor' contains the SQL code for the stored procedure. The bottom 'Results' pane shows the output of the procedure execution, including a table of customer data and a log of actions.

CustomerFirstName	CustomerLastName	CustomerPhoneNo	CustomerEmail	street	city	postalCode
Nyssa	Vazquez	9472785929	nyssa@gmail.com	Lake street	Kingsport	56618

Time	Action	Response	Duration / Fetch Time
194 09:33:40	Call DMDDProject4.GetCustomerFeedbackFor...	1 row(s) returned	0.0050 sec / 0.00002...
195 09:46:02	Apply changes to getCustomerInformation	Changes applied	
196 09:46:15	Call DMDDProject4.getCustomerInformation(5)	Error Code: 1054. Unknown column 'CustomerEmail' in 'field list'	0.00082 sec
197 09:46:29	Apply changes to getCustomerInformation	Changes applied	
198 09:46:34	Call DMDDProject4.getCustomerInformation(5)	1 row(s) returned	0.0012 sec / 0.00002...
199 09:52:01	Call DMDDProject4.getCustomerInformation(7)	1 row(s) returned	0.00048 sec / 0.000...

3)

NAME: GetPaymentStatusWith

This procedure takes a order ID as an input parameter and outputs Payment Status, Payment Method, Payment Date, Payment Time, Payment Error, order Time, fulfilment Status form Order Details, Payment, Order table.

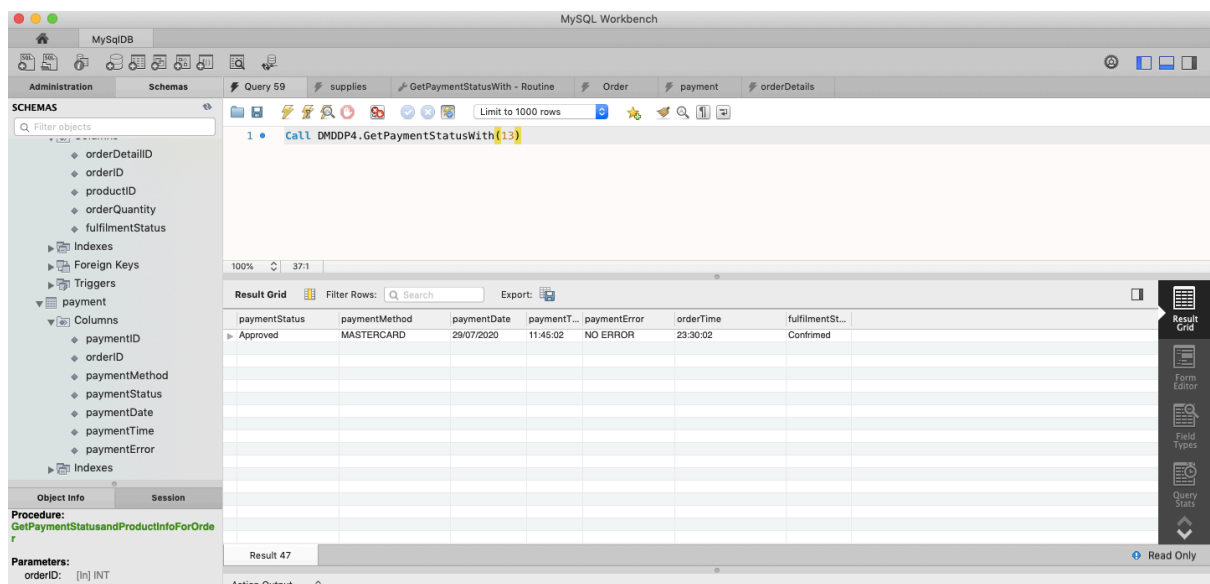
CODE

```
CREATE PROCEDURE `GetPaymentStatusWith`(In order_ID INT)
BEGIN
    SELECT paymentStatus, paymentMethod, paymentDate, paymentTime,
    paymentError, orderTime, fulfilmentStatus
    FROM DMDDP4.orderDetails, DMDDP4.payment, DMDDP4.Order
    WHERE orderDetails.orderID = order_ID and payment.orderID = order_ID and
    DMDDP4.Order.orderID = order_ID;
END
```

EXECUTION

CALL DMDDP4.GetPaymentStatusWith(13)

OUTPUT



MySQL Workbench

Query 59: DMDDP4.GetPaymentStatusWith(13)

Result Grid

paymentStatus	paymentMethod	paymentDate	paymentTime	paymentError	orderTime	fulfilmentStatus
Approved	MASTERCARD	29/07/2020	11:45:02	NO ERROR	23:30:02	Confirmed

Object Info

Procedure: GetPaymentStatusWithProductInfoForOrder

Parameters: orderID: [In] INT

4)

NAME: updateProductPrice

This procedure takes a product ID and the product Price as input parameters and outputs product Name, product Price, product Description, units In Stock and units In Order for the product form Product, Product Stock tables.

This stored procedure calls A TRIGGER.

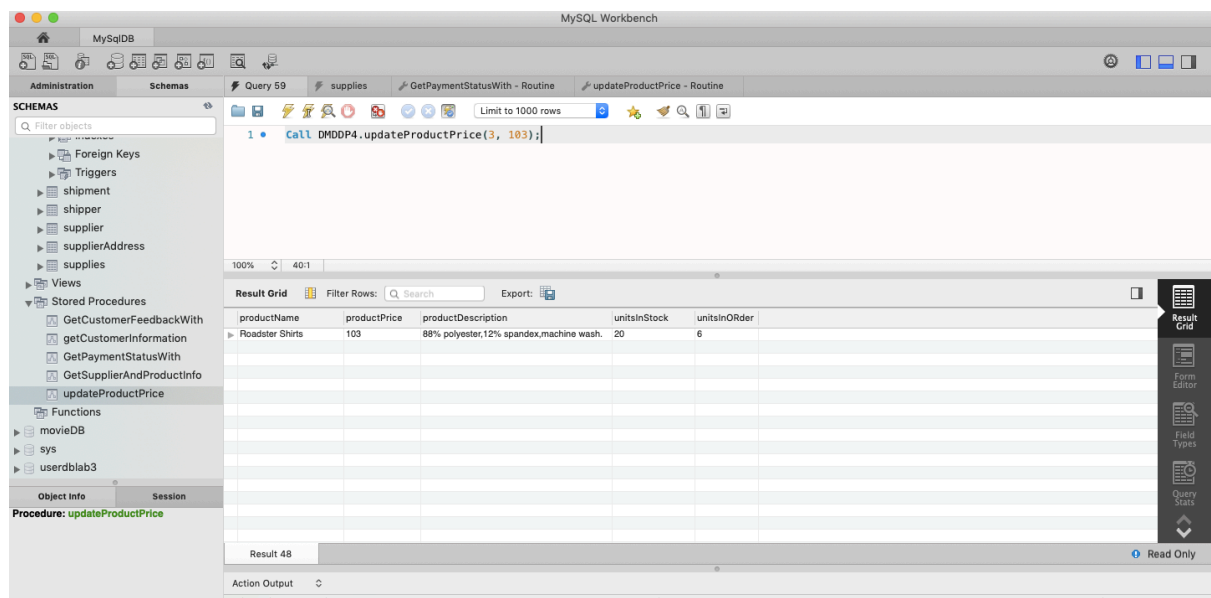
CODE:

```
CREATE `PROCEDURE` `updateProductPrice` (IN product_ID INT, IN product_Price  
VARCHAR(10))  
BEGIN  
    DECLARE currProductPrice VARCHAR(10);  
    SET currProductPrice = (SELECT productPrice from product where productID =  
product_ID);  
    Update product SET productPrice = product_Price where productID = product_ID;  
    SELECT productName, productPrice, productDescription, unitsInStock,  
unitsInORder FROM DMDDP4.product,  
    DMDDP4.productStock WHERE productStock.productID = product_ID and  
product.productID = product_ID;  
END
```

EXECUTION:

Call DMDDP4.updateProductPrice(3, 103);

OUTPUT:



MySQL Workbench interface showing the execution of a stored procedure. The query editor displays the call: `Call DMDDP4.updateProductPrice(3, 103);`. The result grid shows the output of the procedure.

productName	productPrice	productDescription	unitsInStock	unitsInOrder
Roadster Shirts	103	88% polyester,12% spandex,machine wash.	20	6

5)

NAME: GetCustomerIDandOrderIDWithExpirenceRating

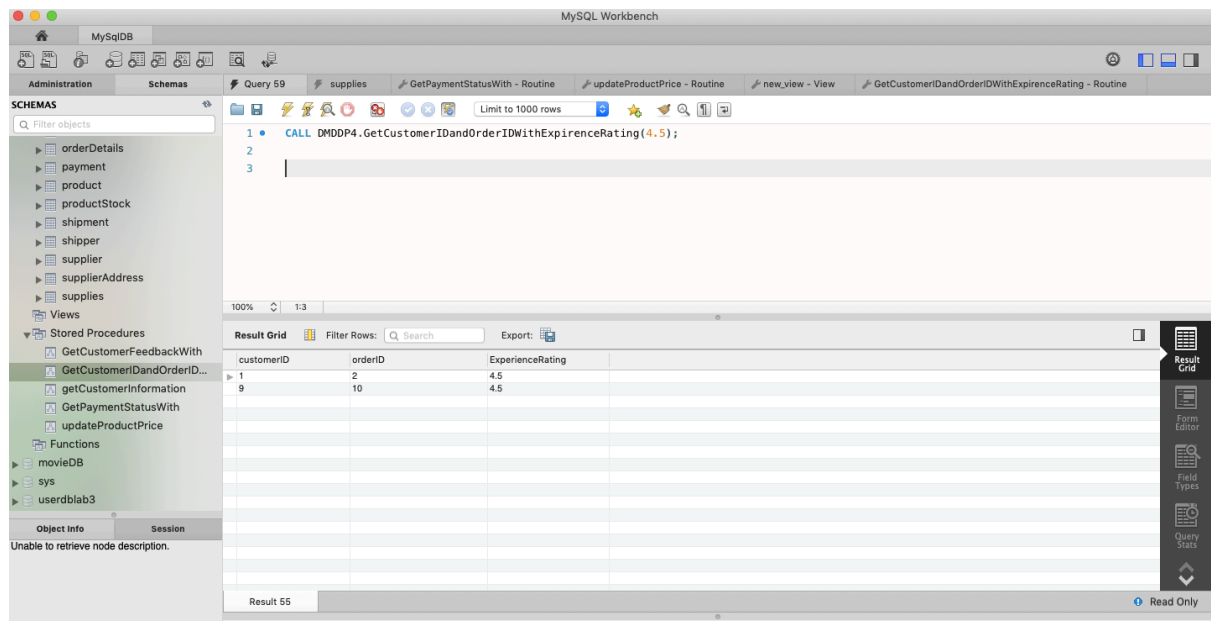
CODE:

```
CREATE PROCEDURE `GetCustomerIDandOrderIDWithExpirenceRating` (In Exp_Rating  
varchar(10))  
BEGIN  
    SELECT customerID, orderID, ExperienceRating FROM  
    DMDDP4.customerFeedback  
    WHERE customerFeedback.ExperienceRating = Exp_Rating;  
END
```

EXECUTION:

CALL DMDDP4.GetCustomerIDandOrderIDWithExpirenceRating(4.5);

OUTPUT:



The screenshot shows the MySQL Workbench interface. The 'Schemas' panel on the left lists various databases, including 'DMDDP4'. The 'Query' window in the center contains the following SQL statement:

```
CALL DMDDP4.GetCustomerIDandOrderIDWithExpirenceRating(4.5);
```

The 'Result Grid' at the bottom displays the output of the query, which is a table with three columns: 'customerID', 'orderID', and 'ExperienceRating'. The table contains two rows of data:

customerID	orderID	ExperienceRating
1	2	4.5
9	10	4.5

The status bar at the bottom indicates 'Result 55' and 'Read Only'.

TRIGGER:

This trigger is called before any update/change of the product price in the product table. It checks for the new product price, if the new product price is less than \$0 then it sets the new product price to 0 and if the product price is more than \$10000 then it sets it to \$10000.

1) CheckProductPriceChange()

```
CREATE TRIGGER `displayProductChanges`  
BEFORE UPDATE ON `product`  
FOR EACH ROW  
BEGIN  
    IF NEW.productPrice < '0' THEN  
        SET NEW.productPrice = '0';  
    ELSEIF NEW.productPrice > '10000' THEN  
        SET NEW.productPrice = '10000';  
    END IF;  
  
END;
```

VIEWS

1)

NAME: CustomersAndTheirCreditCards

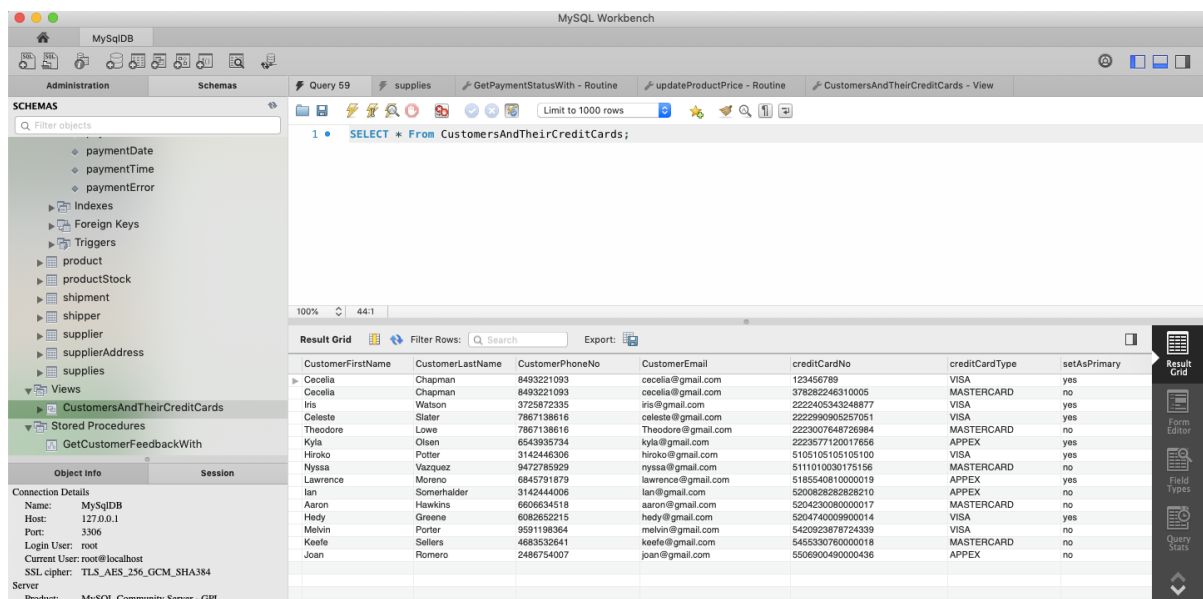
CODE:

```
CREATE VIEW `CustomersAndTheirCreditCards` AS
Select CustomerFirstName, CustomerLastName, CustomerPhoneNo, CustomerEmail,
creditCardNo, creditCardType, setAsPrimary, cardExpiry
FROM DMDDP4.Customer, DMDDP4.creditCard
WHERE Customer.customerID = creditCard.customerID;
```

EXECUTION:

SELECT * From CustomersAndTheirCreditCards;

OUTPUT:



The screenshot shows the MySQL Workbench interface. The Schemas pane on the left lists various database objects, including Views, Stored Procedures, and Tables. The Query Editor in the center contains the SQL code for creating the view and querying it. The Result Grid at the bottom displays the output of the query, showing a list of customers and their credit card details.

CustomerFirstName	CustomerLastName	CustomerPhoneNo	CustomerEmail	creditCardNo	creditCardType	setAsPrimary
Cecelia	Chapman	8493221093	cecelia@gmail.com	123456789	VISA	yes
Cecelia	Chapman	8493221093	cecelia@gmail.com	37682246310005	MASTERCARD	no
Iris	Watson	3725872335	iris@gmail.com	222405343248877	VISA	yes
Celeste	Slater	7867138616	celeste@gmail.com	2222990905257051	VISA	yes
Theodore	Lowe	7867138616	Theodore@gmail.com	2223007648726984	MASTERCARD	no
Kyla	Olson	6543935734	kyla@gmail.com	2223577120017656	APPEX	yes
Hiroko	Potter	3142446306	hiroko@gmail.com	5105105105105100	VISA	yes
Nyssa	Vazquez	9472785929	nyssa@gmail.com	511010030175156	MASTERCARD	no
Lawrence	Moreno	6845791879	lawrence@gmail.com	5185540810000019	APPEX	yes
Ian	Somerhalder	3142444006	ian@gmail.com	5200826282828210	APPEX	no
Aaron	Hawkins	6606934518	aaron@gmail.com	5204230800000017	MASTERCARD	no
Hedy	Greene	6082652215	hedy@gmail.com	5204740009900014	VISA	yes
Melvin	Porter	9591198364	melvin@gmail.com	5420923878724339	VISA	no
Keele	Sellers	4683532641	keefe@gmail.com	5455330760000018	MASTERCARD	no
Joan	Romero	2486754007	joan@gmail.com	55069004900000436	APPEX	no

2)

NAME: ShowSupplierInfoAndTheirAddress

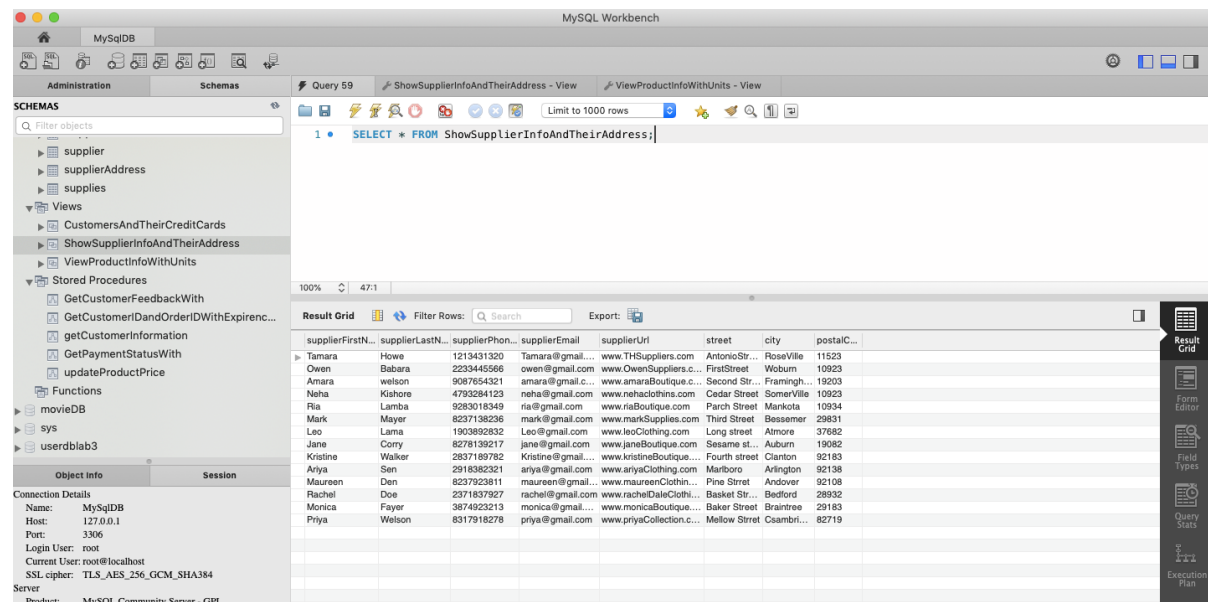
CODE

```
CREATE VIEW `ShowSupplierInfoAndTheirAddress` AS
SELECT supplierFirstName, supplierLastName, supplierPhoneNo, supplierEmail,
supplierUrl, street, city, postalCode
FROM DMDDP4.supplier, DMDDP4.supplierAddress
WHERE supplier.supplierID = supplierAddress.supplierID;
```

EXECUTION:

```
SELECT * From ShowSupplierInfoAndTheirAddress;
```

OUTPUT:



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' tree with 'supplier' and 'supplierAddress' tables expanded. The main query editor shows the query: `SELECT * FROM ShowSupplierInfoAndTheirAddress;`. The 'Result Grid' at the bottom displays the query results in a table format with columns: supplierFirstN..., supplierLastN..., supplierPhon..., supplierEmail, supplierUri, street, city, postalC... The table contains 10 rows of data.

supplierFirstN...	supplierLastN...	supplierPhon...	supplierEmail	supplierUri	street	city	postalC...
Tamara	Howe	1213431320	Tamara@gmail...	www.THSuppliers.com	Antonio Str...	Roseville	11523
Owen	Babara	2233445566	owen@gmail.com	www.OwenSuppliers.c...	First Street	Webum	10923
Amara	wilson	9087654321	amara@gmail.c...	www.amaraBoutique.c...	Second Str...	Framingh...	19203
Neha	Kishore	4793284123	neha@gmail.com	www.nehaClothins.c...	Cedar Street	SomerVile	10923
Ria	Lamba	9083018349	ria@gmail.com	www.riaBoutique.com	Parch Street	Markota	10934
Mark	Mayer	8237198236	mark@gmail.com	www.markSupplies.com	Third Street	Bessemer	29631
Leo	Lama	1903892632	Leo@gmail.com	www.leoClothing.com	Long street	Atmore	37682
Jane	Corry	8278139217	jane@gmail.com	www.janeBoutique...	Sesame st...	Auburn	19082
Kristine	Walker	2837189782	Kristine@gmail...	www.kristineBoutique...	Fourth street	Clanton	92183
Ariya	Sen	2918382321	ariya@gmail.com	www.ariyaClothing.com	Marlboro	Arlington	92138
Maureen	Den	8237923811	maureen@gmail...	www.maureenClothin...	Pine Street	Andover	92108
Rachel	Doe	2371837927	rachel@gmail.com	www.rachelDaleClothi...	Basket Str...	Bedford	28932
Monica	Fayer	3874923213	monica@gmail...	www.monicaBoutique...	Baker Street	Brantree	29183
Priya	Wilson	8317918278	priya@gmail.com	www.priyaCollection.c...	Mellow Street	Csambri...	82719

3)

NAME: ViewProductInfoWithUnits

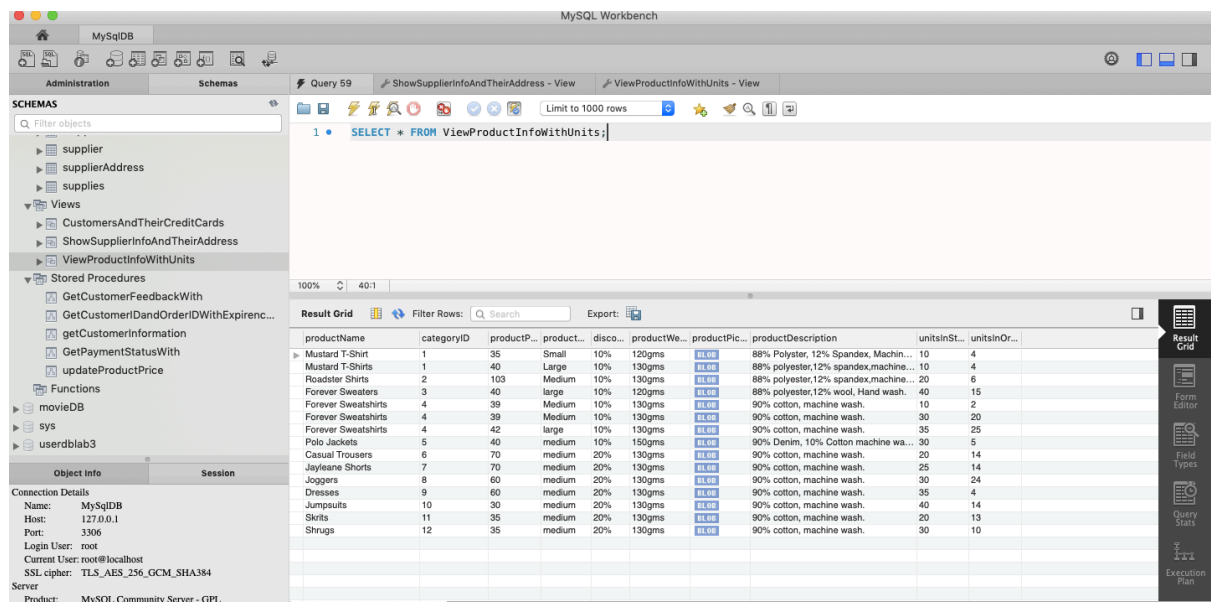
CODE:

```
CREATE VIEW `ViewProductInfoWithUnits` AS
SELECT productName, categoryID, productPrice, productSize, discount, productWeight,
productPicture, productDescription,
unitsInStock, unitsInOrder
FROM DMDDP4.product, DMDDP4.productStock
WHERE product.productID = productStock.productID;
```

EXECUTION:

SELECT * From ViewProductInfoWithUnits;

OUTPUT:


















MySQL Workbench

Administration Schemas Query 59 ShowSupplierInfoAndTheirAddress - View ViewProductInfoWithUnits - View

Limit to 1000 rows

1 • SELECT * FROM ViewProductInfoWithUnits;

100% 40:1

productName	categoryID	productP...	product...	disco...	productWe...	productPic...	productDescription	unitsInSt...	unitsInOr...
Mustard T-Shirt	1	35	Small	10%	120gms		88% Polyester, 12% Spandex, Machin...	10	4
Mustard T-Shirts	1	40	Large	10%	130gms		88% polyester, 12% spandex, machine...	10	4
Roadster Shirts	2	103	Medium	10%	130gms		88% polyester, 12% spandex, machine...	20	6
Forever Sweaters	3	40	large	10%	120gms		88% polyester, 12% wool, Hand wash.	40	15
Forever Sweatshirts	4	39	Medium	10%	130gms		90% cotton, machine wash.	10	2
Forever Sweatshirts	4	39	Medium	10%	130gms		90% cotton, machine wash.	30	20
Forever Sweatshirts	4	42	large	10%	130gms		90% cotton, machine wash.	35	25
Polo Jackets	5	40	medium	10%	150gms		90% Denim, 10% Cotton machine wa...	30	5
Casual Trousers	6	70	medium	20%	130gms		90% cotton, machine wash.	20	14
Jaylene Shorts	7	70	medium	20%	130gms		90% cotton, machine wash.	25	14
Joggers	8	60	medium	20%	130gms		90% cotton, machine wash.	30	24
Dresses	9	60	medium	20%	130gms		90% cotton, machine wash.	35	4
Jumpsuits	10	30	medium	20%	130gms		90% cotton, machine wash.	40	14
Skirts	11	35	medium	20%	130gms		90% cotton, machine wash.	20	13
Strugs	12	35	medium	20%	130gms		90% cotton, machine wash.	30	10

Connection Details

Name: MySQLDB
Host: 127.0.0.1
Port: 3306
Login User: root
Current User: root@localhost
SSL cipher: TLS_AES_256_GCM_SHA384

Server

Product: MySQL Community Server - GPL