Shoe Shopping

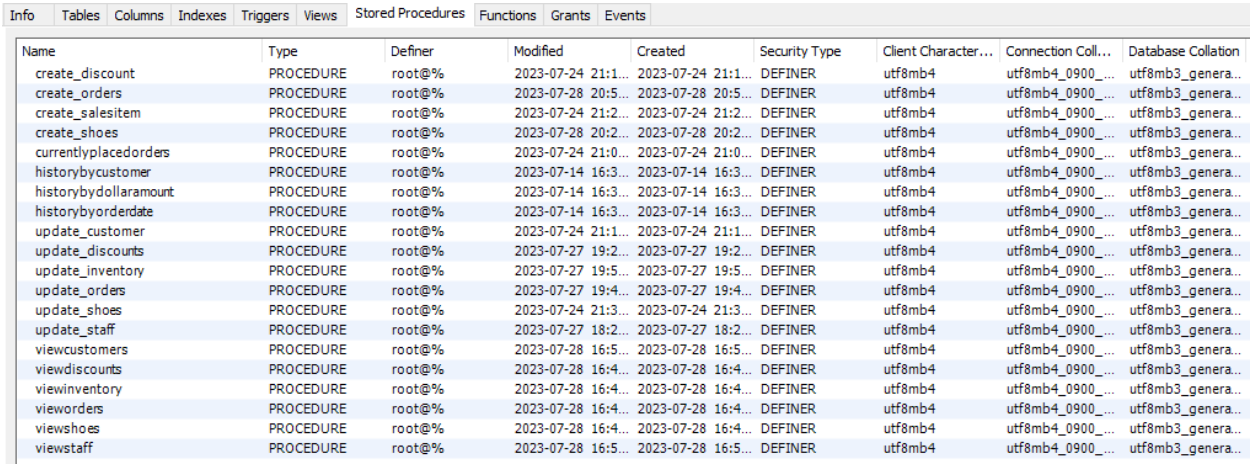
Database User's Guide

Author: Nicholas Ray

Welcome to the Data Dynasty Database User's Guide. The purpose of this document is to display the database's built-in functionality that has been implemented using stored procedures. The website code simply needs to call these stored procedures to add/edit/receive the stored information. Each procedure below has the code used to generate it, a call to that procedure to show how it should be used, and the results of that call. (Both before and after)

All functionality has been created to match the project deliverables/requirements given to us at the beginning of the project. These were tested on a personal copy of the database, as well as the official cloud hosted database the website pulls from.

Stored Procedures



Info | Tables | Columns | Indexes | Triggers | Views | Stored Procedures | Functions | Grants | Events

Name	Type	Definer	Modified	Created	Security Type	Client Character...	Connection Coll...	Database Collation
create_discount	PROCEDURE	root@%	2023-07-24 21:1...	2023-07-24 21:1...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
create_orders	PROCEDURE	root@%	2023-07-28 20:5...	2023-07-28 20:5...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
create_salesitem	PROCEDURE	root@%	2023-07-24 21:2...	2023-07-24 21:2...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
create_shoes	PROCEDURE	root@%	2023-07-28 20:2...	2023-07-28 20:2...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
currentlyplacedorders	PROCEDURE	root@%	2023-07-24 21:0...	2023-07-24 21:0...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
historybycustomer	PROCEDURE	root@%	2023-07-14 16:3...	2023-07-14 16:3...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
historybydollaramount	PROCEDURE	root@%	2023-07-14 16:3...	2023-07-14 16:3...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
historybyorderdate	PROCEDURE	root@%	2023-07-14 16:3...	2023-07-14 16:3...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
update_customer	PROCEDURE	root@%	2023-07-24 21:1...	2023-07-24 21:1...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
update_discounts	PROCEDURE	root@%	2023-07-27 19:2...	2023-07-27 19:2...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
update_inventory	PROCEDURE	root@%	2023-07-27 19:5...	2023-07-27 19:5...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
update_orders	PROCEDURE	root@%	2023-07-27 19:4...	2023-07-27 19:4...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
update_shoes	PROCEDURE	root@%	2023-07-24 21:3...	2023-07-24 21:3...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
update_staff	PROCEDURE	root@%	2023-07-27 18:2...	2023-07-27 18:2...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
viewcustomers	PROCEDURE	root@%	2023-07-28 16:5...	2023-07-28 16:5...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
viewdiscounts	PROCEDURE	root@%	2023-07-28 16:4...	2023-07-28 16:4...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
viewinventory	PROCEDURE	root@%	2023-07-28 16:4...	2023-07-28 16:4...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
vieworders	PROCEDURE	root@%	2023-07-28 16:4...	2023-07-28 16:4...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
viewshoes	PROCEDURE	root@%	2023-07-28 16:4...	2023-07-28 16:4...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...
viewstaff	PROCEDURE	root@%	2023-07-28 16:5...	2023-07-28 16:5...	DEFINER	utf8mb4	utf8mb4_0900_...	utf8mb3_genera...

- create_discount(Percentage)

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
DELIMITER //
CREATE PROCEDURE `create_discount`(IN p_Name VARCHAR(8), IN p_Percentage float)
BEGIN
    INSERT INTO discounts(Name, Percentage) values (p_Name, p_Percentage);
END
//
```

```
CALL create_discount('AC', 0.8);
```

discounts_ID	Percentage	Name
1	0	A
2	0.15	B
3	0.25	C
4	0.35	D
5	0.45	E
6	0.55	AA
7	0.75	AB
NULL	NULL	NULL

discounts_ID	Percentage	Name
1	0	A
2	0.15	B
3	0.25	C
4	0.35	D
5	0.45	E
6	0.55	AA
7	0.75	AB
8	0.8	AC
NULL	NULL	NULL

- create_orders(OrderDate, Units, TotalAmount, Inventory_inv_ID, staff_staff_ID, customers_cust_ID, discounts_discounts_ID)

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
DELIMITER //
CREATE PROCEDURE `create_orders`(IN p_OrderDate DATETIME, IN p_Units INT, IN p_TotalAmount FLOAT,
                                 IN p_Inventory_inv_ID INT, IN p_staff_staff_ID INT,
                                 IN p_customers_cust_ID INT, IN p_discounts_discounts_ID INT)
BEGIN
    SET FOREIGN_KEY_CHECKS=0;
    INSERT INTO orders(OrderDate, OrderStatus, Units, TotalAmount, Inventory_inv_ID, staff_staff_ID, customers_cust_ID, discounts_discounts_ID)
        values(p_OrderDate, 0, p_Units, p_TotalAmount, p_Inventory_inv_ID, p_staff_staff_ID, p_customers_cust_ID, p_discounts_discounts_ID);
    SET FOREIGN_KEY_CHECKS=1;
END
//
```

```
CALL create_orders('2020-11-14 09:50:12', 1, 100.45, 2, 7, 4, 0);
```

order_ID	OrderDate	OrderStatus	Units	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
22	2020-12-19 08:16:11	0	5	462.31	2	5	10	0
21	2020-12-03 23:16:12	0	9	1000.45	3	2	4	4
20	2020-11-14 09:50:12	0	2	175.15	4	10	8	4
19	2020-08-20 14:12:05	0	4	305.55	5	1	5	1
18	2020-07-06 11:20:44	0	6	590	2	6	9	2
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

order_ID	OrderDate	OrderStatus	Units	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
23	2020-11-14 09:50:12	0	1	100.45	2	7	4	0
22	2020-12-19 08:16:11	0	5	462.31	2	5	10	0
21	2020-12-03 23:16:12	0	9	1000.45	3	2	4	4
20	2020-11-14 09:50:12	0	2	175.15	4	10	8	4
19	2020-08-20 14:12:05	0	4	305.55	5	1	5	1
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- create_salesitem(Quantity, shoes_ID);

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
DELIMITER //
CREATE PROCEDURE create_salesitem(IN p_Quantity int, IN p_Shoes_shoes_ID int)
BEGIN
    SET FOREIGN_KEY_CHECKS=0;
    INSERT INTO inventory(SaleStatus, Quantity, Shoes_shoes_ID) values (1, p_Quantity, p_Shoes_shoes_ID);
    SET FOREIGN_KEY_CHECKS=1;
END
//
```

```
CALL create_salesitem(20,5);
```

inv_ID	SaleStatus	Quantity	Shoes_shoes_ID
1	0	10	1
2	0	10	2
3	0	15	3
4	0	10	4
5	0	8	5
6	1	11	1
7	1	5	2
NULL	NULL	NULL	NULL

inv_ID	SaleStatus	Quantity	Shoes_shoes_ID
1	0	10	1
2	0	10	2
3	0	15	3
4	0	10	4
5	0	8	5
6	1	11	1
7	1	5	2
8	1	20	5
NULL	NULL	NULL	NULL

- create_shoes(Name, Brand, Size, Color, Price, Image)

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
DELIMITER //
CREATE PROCEDURE create_shoes(IN p_Name VARCHAR(80), IN p_Brand VARCHAR(25), IN p_Size INT,
                              IN p_Color VARCHAR(15), IN p_Price FLOAT, IN p_Image BLOB)
BEGIN
    SET FOREIGN_KEY_CHECKS=0;
    INSERT INTO shoes(Name, Brand, Size, Color, Price, Image) values(p_Name, p_Brand, p_Size, p_Color, p_price, p_Image);
    SET FOREIGN_KEY_CHECKS=1;
END
//
```

```
CALL create_shoes('NanoX3', 'Reebok', 5, 'Blue', 75.99, '');
```

shoes_ID	Name	Brand	Size	Color	Price	Image
1	Chuck 70	Converse	5	Red	85.55	BLOB
2	FuelCell	New Balance	7	Blue	75.99	BLOB
3	UltraBoost	Adidas	11	Black	105.55	BLOB
4	ThunderCats	Puma	9	White	59.99	BLOB
5	Air Jordon	Nike	8	Green	120	BLOB
NULL	NULL	NULL	NULL	NULL	NULL	NULL

shoes_ID	Name	Brand	Size	Color	Price	Image
1	Chuck 70	Converse	5	Red	85.55	BLOB
2	FuelCell	New Balance	7	Blue	75.99	BLOB
3	UltraBoost	Adidas	11	Black	105.55	BLOB
4	ThunderCats	Puma	9	White	59.99	BLOB
5	Air Jordon	Nike	8	Green	120	BLOB
6	NanoX3	Reebok	5	Blue	75.99	BLOB
NULL	NULL	NULL	NULL	NULL	NULL	NULL

- currentlyplacedorders()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
1    DELIMITER //
2    CREATE PROCEDURE currentlyplacedorders()
3    BEGIN
4        SELECT * FROM orders
5        WHERE OrderStatus = 1;
6    END
7    //
```

```
1    CALL currentlyplacedorders();
```

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

order_ID	OrderDate	OrderStatus	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
11	2020-01-18 06:50:07	1	1475.45	1	5	1	3
12	2020-02-04 10:48:32	1	85.25	2	3	6	4
13	2020-02-17 13:12:44	1	613.99	3	8	3	4
14	2020-03-12 09:19:47	1	219.96	4	9	2	2
15	2020-04-09 04:23:01	1	96.75	1	2	7	0
17	2020-05-23 04:06:52	1	154.15	3	4	4	3

- histroybycustomer()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
1   DELIMITER //
2   CREATE PROCEDURE historybycustomer()
3   BEGIN
4       SELECT * FROM orders
5       WHERE OrderStatus = 0
6       ORDER BY customers_cust_ID;
7   END
8   //
```

```
1   CALL historybycustomer();
```

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

order_ID	OrderDate	OrderStatus	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
4	2019-12-02 09:13:55	0	75.55	4	4	1	0
4	2019-12-08 07:43:22	0	360	1	3	2	1
1	2019-10-29 14:56:59	0	300.55	1	1	3	0
5	2019-12-11 18:30:21	0	1242.95	5	7	4	0
2	2019-11-12 10:32:10	0	105.55	2	2	5	0
8	2019-12-24 15:23:07	0	173.45	5	3	5	4
6	2019-12-21 12:06:57	0	212.56	3	6	6	2
10	2020-01-25 20:42:19	0	966.41	1	10	6	1
7	2019-12-24 09:40:13	0	90.96	2	8	7	0
3	2019-11-24 13:01:45	0	540.45	3	2	8	0
9	2020-01-13 09:51:39	0	75.33	4	1	10	0

- historybydollaramount()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
1    DELIMITER //
2    CREATE PROCEDURE historybydollaramount()
3    BEGIN
4        SELECT * FROM orders
5        WHERE OrderStatus = 0
6        ORDER BY TotalAmount DESC;
7    END
8    //
```

```
1    CALL historybydollaramount();
```

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

order_ID	OrderDate	OrderStatus	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
5	2019-12-11 18:30:21	0	1242.95	5	7	4	0
10	2020-01-25 20:42:19	0	966.41	1	10	6	1
3	2019-11-24 13:01:45	0	540.45	3	2	8	0
4	2019-12-08 07:43:22	0	360	1	3	2	1
1	2019-10-29 14:56:59	0	300.55	1	1	3	0
6	2019-12-21 12:06:57	0	212.56	3	6	6	2
8	2019-12-24 15:23:07	0	173.45	5	3	5	4
2	2019-11-12 10:32:10	0	105.55	2	2	5	0
7	2019-12-24 09:40:13	0	90.96	2	8	7	0
4	2019-12-02 09:13:55	0	75.55	4	4	1	0
9	2020-01-13 09:51:39	0	75.33	4	1	10	0

- historybyorderdate()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
1   DELIMITER //
2   CREATE PROCEDURE historybyorderdate()
3   BEGIN
4       SELECT * FROM orders
5       WHERE OrderStatus = 0
6       ORDER BY OrderDate DESC;
7   END
8   //
```

```
1   CALL historybyorderdate();
```

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

order_ID	OrderDate	OrderStatus	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
10	2020-01-25 20:42:19	0	966.41	1	10	6	1
9	2020-01-13 09:51:39	0	75.33	4	1	10	0
8	2019-12-24 15:23:07	0	173.45	5	3	5	4
7	2019-12-24 09:40:13	0	90.96	2	8	7	0
6	2019-12-21 12:06:57	0	212.56	3	6	6	2
5	2019-12-11 18:30:21	0	1242.95	5	7	4	0
4	2019-12-08 07:43:22	0	360	1	3	2	1
4	2019-12-02 09:13:55	0	75.55	4	4	1	0
3	2019-11-24 13:01:45	0	540.45	3	2	8	0
2	2019-11-12 10:32:10	0	105.55	2	2	5	0
1	2019-10-29 14:56:59	0	300.55	1	1	3	0

- update_customer(cust_ID, C_FirstName, C_LastName, C_Address, C_Email, C_DoB)

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
DELIMITER //
CREATE PROCEDURE update_customer(IN p_cust_ID INT, IN p_C_FirstName VARCHAR(45),
                                 IN p_C_LastName VARCHAR(45), IN p_C_Address VARCHAR(100),
                                 IN p_C_Email VARCHAR(80), IN p_C_DoB DATE)
BEGIN
    Update customers
    SET C_FirstName = COALESCE(p_C_FirstName, C_FirstName),
        C_LastName = COALESCE(p_C_LastName, C_LastName),
        C_Address = COALESCE(p_C_Address, C_Address),
        C_Email = COALESCE(p_C_Email, C_Email),
        C_DoB = COALESCE(p_C_DoB, C_DoB)
    WHERE cust_ID = p_cust_ID;
END
//
```

```
CALL update_customer(1, NULL, 'Yuben', NULL, NULL, NULL);
```

cust_ID	C_FirstName	C_LastName	C_Address	C_Email	C_DoB
1	Abel	Yuqen	45875 Brightbridge Street	SonofYuqen@gmail.com	1990-11-26
2	Franz	Reti	32685 Cycle Circle	FranzTheGreat@gmail.com	1981-03-14
3	Ruben	Thanada	12358 Enduring Path	ThanaBanana@yahoo.com	1994-01-21
4	Alicia	Kant	48981 Oaks Canyon	MrsKant@gmail.com	1966-09-14
5	Manlia	Alam	74336 Bench Street	ListingAlarm@yahoo.com	1974-10-24
6	Erato	Gupta	81131 Market Avenue	EGLLC@gmail.com	1990-05-23
7	Jasos	Khalil	4587 Lincoln Drive	BestJasos@gmail.com	1984-01-04
8	Edgar	Idwal	68122 Open Forest	MoonShoe@yahoo.com	1988-03-22
9	Gi	Severina	77988 Column Court	GiGi123@gmail.com	1974-12-02
10	Livia	Yusuf	15742 Sinking Sea	MsLiv@yahoo.com	1974-08-14
NULL	NULL	NULL	NULL	NULL	NULL

cust_ID	C_FirstName	C_LastName	C_Address	C_Email	C_DoB
1	Abel	Yuben	45875 Brightbridge Street	SonofYuqen@gmail.com	1990-11-26
2	Franz	Reti	32685 Cycle Circle	FranzTheGreat@gmail.com	1981-03-14
3	Ruben	Thanada	12358 Enduring Path	ThanaBanana@yahoo.com	1994-01-21
4	Alicia	Kant	48981 Oaks Canyon	MrsKant@gmail.com	1966-09-14
5	Manlia	Alam	74336 Bench Street	ListingAlarm@yahoo.com	1974-10-24
6	Erato	Gupta	81131 Market Avenue	EGLLC@gmail.com	1990-05-23
7	Jasos	Khalil	4587 Lincoln Drive	BestJasos@gmail.com	1984-01-04
8	Edgar	Idwal	68122 Open Forest	MoonShoe@yahoo.com	1988-03-22
9	Gi	Severina	77988 Column Court	GiGi123@gmail.com	1974-12-02
10	Livia	Yusuf	15742 Sinking Sea	MsLiv@yahoo.com	1974-08-14
NULL	NULL	NULL	NULL	NULL	NULL

- update_discounts(discounts_ID, Percentage)

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
DELIMITER //
CREATE PROCEDURE `update_discounts`(IN p_discounts_ID INT, IN p_Name VARCHAR(8), IN p_Percentage FLOAT)
BEGIN
    Update discounts
    SET Percentage = COALESCE(p_Percentage, Percentage),
        Name = COALESCE(p_Name, Name)
    WHERE discounts_ID = p_discounts_ID;
END
//
```

```
CALL update_discounts(2, NULL, 0.10);
```

discounts_ID	Percentage	Name
1	0	A
2	0.15	B
3	0.25	C
4	0.35	D
5	0.45	E
6	0.55	AA
7	0.75	AB
8	0.8	AC
NULL	NULL	NULL

discounts_ID	Percentage	Name
1	0	A
2	0.1	B
3	0.25	C
4	0.35	D
5	0.45	E
6	0.55	AA
7	0.75	AB
8	0.8	AC
NULL	NULL	NULL

- update_inventory(inv_ID, SaleStatus, Quantity, Shoes_shoes_ID)

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
DELIMITER //
CREATE PROCEDURE update_inventory(IN p_inv_ID INT, IN p_SaleStatus INT,
                                  IN p_Quantity INT, IN p_Shoes_shoes_ID INT)
BEGIN
    SET FOREIGN_KEY_CHECKS=0;
    Update orders
    SET SaleStatus = COALESCE(p_SaleStatus,SaleStatus),
        Quantity = COALESCE(p_Quantity,Quantity),
        Shoes_shoes_ID = COALESCE(p_Shoes_shoes_ID,Shoes_shoes_ID)
    WHERE inv_ID = p_inv_ID;
    SET FOREIGN_KEY_CHECKS=1;
END
//
```

```
CALL update_inventory(1,1,NULL,NULL);
```

inv_ID	SaleStatus	Quantity	Shoes_shoes_ID
1	0	10	1
2	0	10	2
3	0	15	3
4	0	10	4
5	0	8	5
6	1	11	1
7	1	5	2
8	1	20	5
NULL	NULL	NULL	NULL

inv_ID	SaleStatus	Quantity	Shoes_shoes_ID
1	1	10	1
2	0	10	2
3	0	15	3
4	0	10	4
5	0	8	5
6	1	11	1
7	1	5	2
8	1	20	5
NULL	NULL	NULL	NULL

- update_orders(order_ID, OrderDate, OrderStatus, Units, TotalAmount, Inventory_inv_ID, staff_staff_ID, customers_cust_ID, discounts_discounts_ID)

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
DELIMITER //
CREATE PROCEDURE `update_orders`(IN p_order_ID INT, IN p_OrderDate DATETIME,
                                 IN p_OrderStatus INT, p_Units INT, IN p_TotalAmount FLOAT,
                                 IN p_Inventory_inv_ID INT, p_staff_staff_ID INT,
                                 IN p_customers_cust_ID INT, IN p_discounts_discounts_ID INT)
BEGIN
    SET FOREIGN_KEY_CHECKS=0;
    Update orders
    SET OrderDate = COALESCE(p_OrderDate,OrderDate),
        OrderStatus = COALESCE(p_OrderStatus,OrderStatus),
        Units = COALESCE(p_Units, Units),
        TotalAmount = COALESCE(p_TotalAmount,TotalAmount),
        Inventory_inv_ID = COALESCE(p_Inventory_inv_ID,Inventory_inv_ID),
        staff_staff_ID = COALESCE(p_staff_staff_ID,staff_staff_ID),
        customers_cust_ID = COALESCE(p_customers_cust_ID,customers_cust_ID),
        discounts_discounts_ID = COALESCE(p_discounts_discounts_ID,discounts_discounts_ID)
    WHERE order_ID = p_order_ID;
    SET FOREIGN_KEY_CHECKS=1;
END
//
```

```
CALL update_orders(2, NULL, 1, NULL, NULL, NULL, NULL, NULL, NULL);
```

order_ID	OrderDate	OrderStatus	Units	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
1	2019-10-29 14:56:59	1	4	300.55	1	1	3	1
2	2019-11-12 10:32:10	0	1	105.55	2	2	5	0
3	2019-11-24 13:01:45	0	6	540.45	3	2	8	0
4	2019-12-08 07:43:22	0	3	360	1	3	2	1
5	2019-12-11 18:30:21	0	8	1242.95	5	7	4	0
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

order_ID	OrderDate	OrderStatus	Units	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
1	2019-10-29 14:56:59	1	4	300.55	1	1	3	1
2	2019-11-12 10:32:10	1	1	105.55	2	2	5	0
3	2019-11-24 13:01:45	0	6	540.45	3	2	8	0
4	2019-12-08 07:43:22	0	3	360	1	3	2	1
5	2019-12-11 18:30:21	0	8	1242.95	5	7	4	0
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- update_shoes(shoes_ID, Name, Brand, Size, Color, Price, Image)

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
1   DELIMITER //
2   CREATE PROCEDURE update_shoes(IN p_shoes_ID INT, IN p_Name VARCHAR(80),
3                                 IN p_Brand VARCHAR(25), IN p_Size INT,
4                                 IN p_Color VARCHAR(15), IN p_Price FLOAT, IN p_Image BLOB)
5   BEGIN
6       Update shoes
7       SET Name = COALESCE(p_Name, Name),
8           Brand = COALESCE(p_Brand, Brand),
9           Size = COALESCE(p_Size, Size),
10          Color = COALESCE(p_Color, Color),
11          Price = COALESCE(p_Price, Price),
12          Image = COALESCE(p_Image, Image)
13      WHERE shoes_ID = p_shoes_ID;
14  END
15  //
```

```
1   CALL update_shoes(1, NULL, NULL, 7, NULL, NULL, NULL);
```

shoes_ID	Name	Brand	Size	Color	Price	Image
1	Chuck 70	Converse	5	Red	85.55	BLOB
2	FuelCell	New Balance	7	Blue	75.99	BLOB
3	UltraBoost	Adidas	11	Black	105.55	BLOB
4	ThunderCats	Puma	9	White	59.99	BLOB
5	Air Jordon	Nike	8	Green	120	BLOB
NULL	NULL	NULL	NULL	NULL	NULL	NULL

shoes_ID	Name	Brand	Size	Color	Price	Image
1	Chuck 70	Converse	7	Red	85.55	BLOB
2	FuelCell	New Balance	7	Blue	75.99	BLOB
3	UltraBoost	Adidas	11	Black	105.55	BLOB
4	ThunderCats	Puma	9	White	59.99	BLOB
5	Air Jordon	Nike	8	Green	120	BLOB
NULL	NULL	NULL	NULL	NULL	NULL	NULL

- update_staff(staff_ID, ST_FirstName, ST_LastName, ST_Address, ST_Email, ST_DoB, ST_Password)

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its before/after in the specified table.

```
DELIMITER //
CREATE PROCEDURE update_staff(IN p_staff_ID INT, IN p_ST_FirstName VARCHAR(45),
                              IN p_ST_LastName VARCHAR(45), IN p_ST_Address VARCHAR(100),
                              IN p_ST_Email VARCHAR(80), IN p_ST_DoB DATE, IN p_ST_Password VARCHAR(100))
BEGIN
    Update staff
    SET ST_FirstName = COALESCE(p_ST_FirstName, ST_FirstName),
        ST_LastName = COALESCE(p_ST_LastName, ST_LastName),
        ST_Address = COALESCE(p_ST_Address,ST_Address),
        ST_Email = COALESCE(p_ST_Email,ST_Email),
        ST_DoB = COALESCE(p_ST_DoB,ST_DoB),
        ST_Password = COALESCE(p_ST_Password,ST_Password)
    WHERE staff_ID = p_staff_ID;
END
//
```

```
CALL update_staff(1,'Charles',NULL,NULL,NULL,NULL,NULL)
```

staff_ID	ST_FirstName	ST_LastName	ST_Address	ST_Email	ST_DoB	ST_Password
1	Charlie	Baker	15648 Bend Street	CBaker@gmail.com	1981-01-19	Password
2	Heather	Wonder	12348 Yule Drive	WonderWoman@gmail.com	1993-04-12	Password
3	David	Brash	75479 Canyon Oaks	BigBrash@yahoo.com	1991-12-30	Password
4	Monica	Liu	13993 Benard Avenue	TheM@gmail.com	2003-06-24	Password
5	Dion	Tempest	11404 State Court	StormingD@yahoo.com	1989-02-13	Password
6	Emily	Vega	361 Maple Oak Drive	EmilyVega@yahoo.com	2001-03-10	Password
7	Momo	Harb	12348 Cinnamon Street	HarbBro@gmail.com	1987-11-15	Password
8	Kasto	Issa	7845 Ember Creek	RollingIssa@yahoo.com	1989-05-07	Password
9	Leah	Tind	6671 Fig Leaf Drive	TindofLife@gmail.com	2005-08-30	Password
10	Nzo	Rubis	71265 Washington Avenue	TopBrassNzo@yahoo.com	1988-07-16	Password
NULL	NULL	NULL	NULL	NULL	NULL	NULL

staff_ID	ST_FirstName	ST_LastName	ST_Address	ST_Email	ST_DoB	ST_Password
1	Charles	Baker	15648 Bend Street	CBaker@gmail.com	1981-01-19	Password
2	Heather	Wonder	12348 Yule Drive	WonderWoman@gmail.com	1993-04-12	Password
3	David	Brash	75479 Canyon Oaks	BigBrash@yahoo.com	1991-12-30	Password
4	Monica	Liu	13993 Benard Avenue	TheM@gmail.com	2003-06-24	Password
5	Dion	Tempest	11404 State Court	StormingD@yahoo.com	1989-02-13	Password
6	Emily	Vega	361 Maple Oak Drive	EmilyVega@yahoo.com	2001-03-10	Password
7	Momo	Harb	12348 Cinnamon Street	HarbBro@gmail.com	1987-11-15	Password
8	Kasto	Issa	7845 Ember Creek	RollingIssa@yahoo.com	1989-05-07	Password
9	Leah	Tind	6671 Fig Leaf Drive	TindofLife@gmail.com	2005-08-30	Password
10	Nzo	Rubis	71265 Washington Avenue	TopBrassNzo@yahoo.com	1988-07-16	Password
NULL	NULL	NULL	NULL	NULL	NULL	NULL

- viewcustomers()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
DELIMITER //
CREATE PROCEDURE viewcustomers()
BEGIN
    SELECT * FROM shoeshopping.customers;
END
//
```

```
CALL viewcustomers();
```

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

cust_ID	C_FirstName	C_LastName	C_Address	C_Email	C_DoB
1	Abel	Yuben	45875 Brightbridge Street	SonofYuqen@gmail.com	1990-11-26
2	Franz	Reti	32685 Cycle Circle	FranzTheGreat@gmail.com	1981-03-14
3	Ruben	Thanada	12358 Enduring Path	ThanaBanana@yahoo.com	1994-01-21
4	Alicia	Kant	48981 Oaks Canyon	MrsKant@gmail.com	1966-09-14
5	Manlia	Alam	74336 Bench Street	ListingAlarm@yahoo.com	1974-10-24
6	Erato	Gupta	81131 Market Avenue	EGLLC@gmail.com	1990-05-23
7	Jasos	Khalil	4587 Lincoln Drive	BestJasos@gmail.com	1984-01-04
8	Edgar	Idwal	22 Open Forest	MoonShoe@yahoo.com	1988-03-22
9	Gi	Severina	77988 Column Court	GiGi123@gmail.com	1974-12-02
10	Livia	Yusuf	15742 Sinking Sea	MsLiv@yahoo.com	1974-08-14

Khalil

- viewdiscounts()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
DELIMITER //
CREATE PROCEDURE viewdiscounts()
BEGIN
    SELECT * FROM shoeshopping.discounts;
END
//
```

```
CALL viewdiscounts();
```

Result Grid | Filter Rows:

discounts_ID	Percentage
1	0
2	0.15
3	0.25
4	0.35
5	0.45
6	0.55
7	0.75

- viewinventory()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
DELIMITER //
CREATE PROCEDURE viewinventory()
BEGIN
    SELECT * FROM shoeshopping.inventory;
END
//
```

```
CALL viewinventory();
```

Result Grid | Filter Rows: | Export

inv_ID	SaleStatus	Quantity	Shoes_shoes_ID
1	1	10	1
2	0	10	2
3	0	15	3
4	0	10	4
5	0	8	5
6	1	11	1
7	1	5	2
8	1	20	5

- vieworders()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
1    DELIMITER //
2    CREATE PROCEDURE vieworders()
3    BEGIN
4        SELECT * FROM shoeshopping.orders;
5    END
6    //
```

```
1    CALL vieworders();
```

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

order_ID	OrderDate	OrderStatus	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
1	2019-10-29 14:56:59	1	300.55	1	1	3	1
2	2019-11-12 10:32:10	0	105.55	2	2	5	0
3	2019-11-24 13:01:45	0	540.45	3	2	8	0
4	2019-12-08 07:43:22	0	360	1	3	2	1
4	2019-12-02 09:13:55	0	75.55	4	4	1	0
5	2019-12-11 18:30:21	0	1242.95	5	7	4	0
6	2019-12-21 12:06:57	0	212.56	3	6	6	2
7	2019-12-24 09:40:13	0	90.96	2	8	7	0
8	2019-12-24 15:23:07	0	173.45	5	3	5	4
9	2020-01-13 09:51:39	0	75.33	4	1	10	0
10	2020-01-25 20:42:19	0	966.41	1	10	6	1
11	2020-01-18 06:50:07	1	1475.45	1	5	1	3
12	2020-02-04 10:48:32	1	85.25	2	3	6	4
13	2020-02-17 13:12:44	1	613.99	3	8	3	4
14	2020-03-12 09:19:47	1	219.96	4	9	2	2
15	2020-04-09 04:23:01	1	96.75	1	2	7	0
16	2020-05-02 19:59:28	0	1313.36	5	7	10	1
17	2020-05-23 04:06:52	1	154.15	3	4	4	3
18	2020-07-06 11:20:44	0	590	2	6	9	2
19	2020-08-20 14:12:05	0	305.55	5	1	5	1
20	2020-11-14 09:50:12	0	175.15	4	10	8	4

- viewshoes()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
1    DELIMITER //
2    CREATE PROCEDURE viewshoes()
3    BEGIN
4        SELECT * FROM shoeshopping.shoes;
5    END
6    //
```

```
1    CALL viewshoes();
```

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

shoes_ID	Name	Brand	Size	Color	Price	Image
1	Chuck 70	Converse	7	Red	85.55	BLOB
2	FuelCell	New Balance	7	Blue	75.99	BLOB
3	UltraBoost	Adidas	11	Black	105.55	BLOB
4	ThunderCats	Puma	9	White	59.99	BLOB
5	Air Jordon	Nike	8	Green	120	BLOB

- viewstaff()

Below is the SQL code to create the stored procedure within the database followed by a call to that procedure and its output.

```
DELIMITER //
CREATE PROCEDURE viewstaff()
BEGIN
    SELECT * FROM shoeshopping.staff;
END
//
```

```
CALL viewstaff();
```

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

staff_ID	ST_FirstName	ST_LastName	ST_Address	ST_Email	ST_DoB	ST_Password
1	Charles	Baker	15648 Bend Street	CBaker@gmail.com	1981-01-19	Password
2	Heather	Wonder	12348 Yule Drive	WonderWoman@gmail.com	1993-04-12	Password
3	David	Brash	75479 Canyon Oaks	BigBrash@yahoo.com	1991-12-30	Password
4	Monica	Liu	13993 Benard Avenue	TheM@gmail.com	2003-06-24	Password
5	Dion	Tempest	11404 State Court	StormingD@yahoo.com	1989-02-13	Password
6	Emily	Vega	361 Maple Oak Drive	EmilyVega@yahoo.com	2001-03-10	Password
7	Momo	Harb	12348 Cinnamon Street	HarbBro@gmail.com	1987-11-15	Password
8	Kasto	Issa	7845 Ember Creek	RollingIssa@yahoo.com	1989-05-07	Password
9	Leah	Tind	6671 Fig Leaf Drive	TindofLife@gmail.com	2005-08-30	Password
10	Nzo	Rubis	71265 Washington Avenue	TopBrassNzo@yahoo.com	1988-07-16	Password

Triggers

- after_shoes_insert()

This is a database trigger that creates a matching inventory row when a new shoe is added to the database. Do not call this, it doesn't work like a stored procedure.

```
DELIMITER //

CREATE TRIGGER after_shoes_insert
AFTER INSERT
ON shoes FOR EACH ROW
BEGIN
    INSERT INTO inventory(SaleStatus, Quantity, Shoes_shoes_ID)
    VALUES (0, 5, NEW.shoes_ID);
END
//
```