# **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**

Triggers Views	Stored Procedures	Functions Grants Ever	nts				
Type	Definer	Modified	Created	Security Type	Client Character	Connection Coll	Database Collation
PROCEDURE	root@%	2023-07-24 21:1	2023-07-24 21:1	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-28 20:5	2023-07-28 20:5	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera
PROCEDURE	root@%	2023-07-24 21:2	2023-07-24 21:2	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-28 20:2	2023-07-28 20:2	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-24 21:0	2023-07-24 21:0	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera
PROCEDURE	root@%	2023-07-14 16:3	2023-07-14 16:3	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-14 16:3	2023-07-14 16:3	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-14 16:3	2023-07-14 16:3	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-24 21:1	2023-07-24 21:1	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-27 19:2	2023-07-27 19:2	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-27 19:5	2023-07-27 19:5	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-27 19:4	2023-07-27 19:4	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-24 21:3	2023-07-24 21:3	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-27 18:2	2023-07-27 18:2	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-28 16:5	2023-07-28 16:5	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera
PROCEDURE	root@%	2023-07-28 16:4	2023-07-28 16:4	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-28 16:4	2023-07-28 16:4	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-28 16:4	2023-07-28 16:4	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-28 16:4	2023-07-28 16:4	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
PROCEDURE	root@%	2023-07-28 16:5	2023-07-28 16:5	DEFINER	utf8mb4	utf8mb4_0900	utf8mb3_genera.
	Type PROCEDURE	Type Definer PROCEDURE root@%	Type Definer Modified PROCEDURE root@% 2023-07-24 21:1 PROCEDURE root@% 2023-07-28 20:5 PROCEDURE root@% 2023-07-24 21:2 PROCEDURE root@% 2023-07-24 21:2 PROCEDURE root@% 2023-07-24 21:0 PROCEDURE root@% 2023-07-14 16:3 PROCEDURE root@% 2023-07-14 16:3 PROCEDURE root@% 2023-07-14 16:3 PROCEDURE root@% 2023-07-14 16:3 PROCEDURE root@% 2023-07-24 21:1 PROCEDURE root@% 2023-07-27 19:2 PROCEDURE root@% 2023-07-27 19:5 PROCEDURE root@% 2023-07-27 19:4 PROCEDURE root@% 2023-07-27 19:4 PROCEDURE root@% 2023-07-28 16:4	Type Definer Modified Created PROCEDURE root@% 2023-07-24 21:1. 2023-07-24 21:1. PROCEDURE root@% 2023-07-28 20:5. 2023-07-28 20:5. PROCEDURE root@% 2023-07-28 20:2. 2023-07-28 20:2. PROCEDURE root@% 2023-07-28 20:2. 2023-07-28 20:2. PROCEDURE root@% 2023-07-24 21:0. 2023-07-24 21:0. PROCEDURE root@% 2023-07-24 21:0. 2023-07-24 21:0. PROCEDURE root@% 2023-07-14 16:3. 2023-07-14 16:3. PROCEDURE root@% 2023-07-14 16:3. 2023-07-14 16:3. PROCEDURE root@% 2023-07-14 16:3. 2023-07-14 16:3. PROCEDURE root@% 2023-07-24 21:1. 2023-07-24 21:1. PROCEDURE root@% 2023-07-27 19:2. 2023-07-24 21:1. PROCEDURE root@% 2023-07-27 19:5. 2023-07-27 19:5. PROCEDURE root@% 2023-07-27 19:4. 2023-07-27 19:4. PROCEDURE root@% 2023-07-27 19:4. 2023-07-27 19:4. PROCEDURE root@% 2023-07-27 18:2. 2023-07-27 18:2. PROCEDURE root@% 2023-07-28 16:4. 2023-07-28 16:4.	Type Definer Modified Created Security Type PROCEDURE root@% 2023-07-24 21:1 2023-07-24 21:1 DEFINER PROCEDURE root@% 2023-07-28 20:5 2023-07-28 20:5 DEFINER PROCEDURE root@% 2023-07-28 20:2 2023-07-24 21:2 DEFINER PROCEDURE root@% 2023-07-28 20:2 2023-07-24 21:2 DEFINER PROCEDURE root@% 2023-07-24 21:0 DEFINER PROCEDURE root@% 2023-07-24 21:0 DEFINER PROCEDURE root@% 2023-07-14 16:3 2023-07-24 1:0 DEFINER PROCEDURE root@% 2023-07-14 16:3 2023-07-14 16:3 DEFINER PROCEDURE root@% 2023-07-24 1:1 2023-07-27 19:2 DEFINER PROCEDURE root@% 2023-07-27 19:2 2023-07-27 19:2 DEFINER PROCEDURE root@% 2023-07-27 19:3 2023-07-27 19:4 DEFINER PROCEDURE root@% 2023-07-27 19:4 2023-07-27 19:4 DEFINER PROCEDURE root@% 2023-07-24 1:3 DEFINER PROCEDURE root@% 2023-07-24 1:3 DEFINER PROCEDURE root@% 2023-07-28 16:4 2023-07-28 16:4 DEFINER PROCEDURE root@% 2023-07-28 16:4 DEFINER PROCEDURE root@% 2023-07-28 16:4 2023-07-28 16:4 DEFINER PROCEDURE	Type Definer Modified Created Security Type Client Character PROCEDURE root@% 2023-07-24 21:1 2023-07-24 21:1 DEFINER utf8mb4 PROCEDURE root@% 2023-07-28 20:5 2023-07-28 20:5 DEFINER utf8mb4 PROCEDURE root@% 2023-07-28 20:2 2023-07-24 21:2 DEFINER utf8mb4 PROCEDURE root@% 2023-07-28 20:2 2023-07-24 21:2 DEFINER utf8mb4 PROCEDURE root@% 2023-07-28 20:2 2023-07-24 21:0 DEFINER utf8mb4 PROCEDURE root@% 2023-07-14 16:3 2023-07-24 21:0 DEFINER utf8mb4 PROCEDURE root@% 2023-07-14 16:3 2023-07-14 16:3 DEFINER utf8mb4 PROCEDURE root@% 2023-07-24 21:1 2023-07-24 21:1 DEFINER utf8mb4 PROCEDURE root@% 2023-07-27 19:2 2023-07-27 19:2 DEFINER utf8mb4 PROCEDURE root@% 2023-07-27 19:4 2023-07-27 19:5 DEFINER utf8mb4 PROCEDURE root@% 2023-07-27 19:4 2023-07-27 19:4 DEFINER utf8mb4 PROCEDURE root@% 2023-07-27 19:4 2023-07-27 19:4 DEFINER utf8mb4 PROCEDURE root@% 2023-07-27 19:4 2023-07-27 19:4 DEFINER utf8mb4 PROCEDURE root@% 2023-07-28 16:4 2023-07-27 19:1 DEFINER utf8mb4 PROCEDURE root@% 2023-07-28 16:5 2023-07-28 16:4 DEFINER utf8mb4 PROCEDURE root@% 2023-07-28 16:4 DEFINER utf8mb4 PROCEDURE root@% 2023-07-28 16:4 2023-07-28 16:4 DEFINER utf8mb4 PROCEDURE roo	Type Definer Modified Created Security Type Client Character Connection Coll PROCEDURE root@% 2023-07-24 21:1. 2023-07-24 21:1. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-28 20:5. 2023-07-28 20:5. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-28 20:2. 2023-07-24 21:2. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-28 20:2. 2023-07-28 20:2. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-24 21:0. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-14 16:3. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-24 11:0. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-24 11:0. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-27 19:2. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-27 19:5. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-27 18:2. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-28 16:5. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-28 16:5. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-28 16:5. DEFINER utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-28 16:4. DEFINER utf8mb4 utf8mb4_0900 Utf8mb4 utf8mb4_0900 PROCEDURE root@% 2023-07-28 16:4. DEFINER utf8mb4 utf8mb4_0900 Utf8mb4 utf8mb4_0900

• 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
//
```

## 1 CALL create\_discount('AC', 0.8);

	discounts_ID	Percentage	Name
•	1	0	Α
	2	0.15	В
	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	Α
	2	0.15	В
	3	0.25	С
	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, 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 //

Pocker Procedure create_orders(IN p_OrderDate DATETIME, 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_ID INT)

BEGIN

SET FOREIGN_KEY_CHECKS=0;

INSERT INTO orders(OrderDate, OrderStatus, TotalAmount, Inventory_inv_ID, staff_staff_ID, customers_cust_ID, discounts_discounts_ID)

values(p_OrderDate, 0, p_TotalAmount, p_Inventory_inv_ID, p_staff_staff_ID, p_customers_cust_ID, p_discounts_discounts_ID);

SET FOREIGN_KEY_CHECKS=1;

END

11 //
```

### 1 • CALL create\_orders('2020-12-03 23:16:12', 1000.45, 3, 2, 4, 4);

	order_ID	OrderDate	OrderStatus	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
•	20	2020-11-14 09:50:12	0	175.15	4	10	8	4
	19	2020-08-20 14:12:05	0	305.55	5	1	5	1
	18	2020-07-06 11:20:44	0	590	2	6	9	2
	17	2020-05-23 04:06:52	1	154.15	3	4	4	3
	16	2020-05-02 19:59:28	0	1313.36	5	7	10	1
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	HULL

	order_ID	OrderDate	OrderStatus	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
•	21	2020-12-03 23:16:12	0	1000.45	3	2	4	4
	20	2020-11-14 09:50:12	0	175.15	4	10	8	4
	19	2020-08-20 14:12:05	0	305.55	5	1	5	1
	18	2020-07-06 11:20:44	0	590	2	6	9	2
	17	2020-05-23 04:06:52	1	154.15	3	4	4	3
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	HULL

• 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

//
```

## 1 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

///
```

1 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

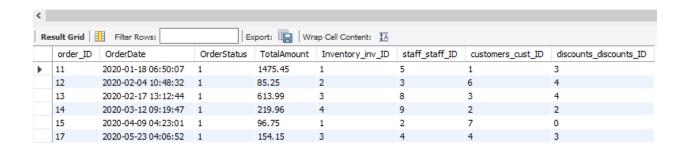
• currentlyplacedorders()

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

```
DELIMITER //
CREATE PROCEDURE currentlyplacedorders()

BEGIN
SELECT * FROM orders
WHERE OrderStatus = 1;
END

///
```



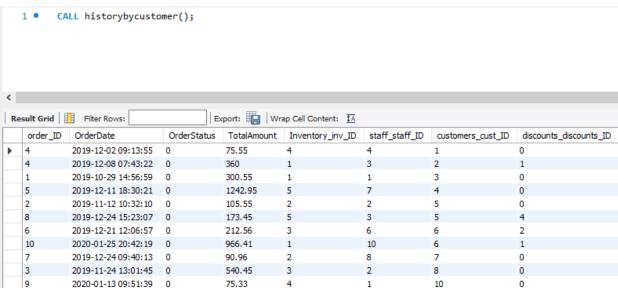
• histroybycustomer()

```
DELIMITER //
CREATE PROCEDURE historybycustomer()

BEGIN

SELECT * FROM orders
WHERE OrderStatus = 0
ORDER BY customers_cust_ID;
END

//
```



• historybydollaramount()

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 historybydollaramount()

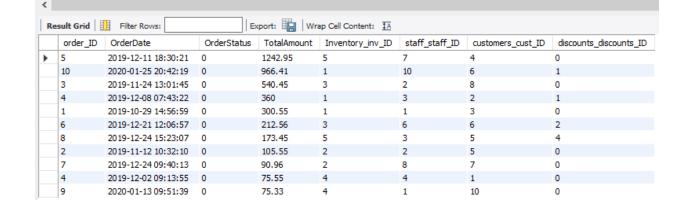
BEGIN

SELECT * FROM orders
WHERE OrderStatus = 0
ORDER BY TotalAmount DESC;

END

//
```

1 • CALL historybydollaramount();



• historybyorderdate()

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 historybyorderdate()

BEGIN

SELECT * FROM orders
WHERE OrderStatus = 0
ORDER BY OrderDate DESC;
END

//
```

CALL historybyorderdate(); Result Grid Filter Rows: Export: Wrap Cell Content: ‡A 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 2020-01-13 09:51:39 0 75.33 1 10 0 2019-12-24 15:23:07 0 173.45 5 5 3 7 2019-12-24 09:40:13 0 90.96 2 8 7 0 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 0 2019-12-08 07:43:22 0 360 3 4 2019-12-02 09:13:55 0 75.55 4 4 1 0 540.45 2 8 0 3 2019-11-24 13:01:45 0 3 2 2019-11-12 10:32:10 0 105.55 2 0 2019-10-29 14:56:59 0 300.55

• 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 //
 2 • ⊖ CREATE PROCEDURE update_customer(IN p_cust_ID INT, IN p_C_FirstName VARCHAR(45),
 3
                                       IN p_C_LastName VARCHAR(45), IN p_C_Address VARCHAR(100),
 4
                                       IN p_C_Email VARCHAR(80), IN p_C_DoB DATE)
 5

⊖ BEGIN

 6
           Update customers
           SET C_FirstName = COALESCE(p_C_FirstName, C_FirstName),
 7
               C_LastName = COALESCE(p_C_LastName, C_LastName),
 8
 9
               C_Address = COALESCE(p_C_Address, C_Address),
               C_Email = COALESCE(p_C_Email, C_Email),
10
               C_DoB = COALESCE(p_C_DoB, C_DoB)
11
           WHERE cust_ID = p_cust_ID;
12
      END
13
14
       //
```

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	HULL	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

//
```

1 • CALL update\_discounts(2, NULL, 0.10);

	discounts_ID	Percentage	Name
•	1	0	Α
	2	0.15	В
	3	0.25	С
	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	Α
	2	0.1	В
	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 //
 1
 2 ● ○ CREATE PROCEDURE update_inventory(IN p_inv_ID INT, IN p_SaleStatus INT,
 3
                                    IN p_Quantity INT, IN p_Shoes_shoes_ID INT)

⊖ BEGIN

4
           SET FOREIGN_KEY_CHECKS=0;
 5
           Update orders
 6
 7
           SET SaleStatus = COALESCE(p_SaleStatus, SaleStatus),
               Quantity = COALESCE(p_Quantity,Quantity),
 8
               Shoes_shoes_ID = COALESCE(p_Shoes_shoes_ID, Shoes_shoes_ID)
9
           WHERE inv_ID = p_inv_ID;
10
           SET FOREIGN_KEY_CHECKS=1;
11
12
      - END
13
       //
```

#### 1 • 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	HULL

• update\_orders(order\_ID, OrderDate, OrderStatus, 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.

```
1
       DELIMITER //
 2 • ○ CREATE PROCEDURE update_orders(IN p_order_ID INT, IN p_OrderDate DATETIME,
 3
                                    IN p_OrderStatus INT, IN p_TotalAmount FLOAT,
                                    IN p_Inventory_inv_ID INT, p_staff_staff_ID INT,
 4
 5
                                    IN p_customers_cust_ID INT, IN p_discounts_discounts_ID INT)

⊖ BEGIN

 7
           SET FOREIGN_KEY_CHECKS=0;
           Update orders
 8
           SET OrderDate = COALESCE(p_OrderDate,OrderDate),
 9
               OrderStatus = COALESCE(p_OrderStatus,OrderStatus),
10
               TotalAmount = COALESCE(p_TotalAmount, TotalAmount),
11
12
               Inventory_inv_ID = COALESCE(p_Inventory_inv_ID,Inventory_inv_ID),
               staff_staff_ID = COALESCE(p_staff_staff_ID),staff_staff_ID),
14
               customers_cust_ID = COALESCE(p_customers_cust_ID), customers_cust_ID),
               discounts_discounts_ID = COALESCE(p_discounts_discounts_ID, discounts_discounts_ID)
15
           WHERE order_ID = p_order_ID;
16
           SET FOREIGN_KEY_CHECKS=1;
17
18
       END
19
       //
```

#### 1 • CALL update orders(1, NULL, 1, NULL, NULL, NULL, 1);

	order_ID	OrderDate	OrderStatus	TotalAmount	Inventory_inv_ID	staff_staff_ID	customers_cust_ID	discounts_discounts_ID
•	1	2019-10-29 14:56:59	0	300.55	1	1	3	0
	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
	NULL	NULL	NULL	NULL	HULL	NULL	NULL	NULL

	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
	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.

```
DELIMITER //
2 • Q CREATE PROCEDURE update_shoes(IN p_shoes_ID INT, IN p_Name VARCHAR(80),
3
                                    IN p_Brand VARCHAR(25), IN p_Size INT,
                                    IN p_Color VARCHAR(15), IN p_Price FLOAT, IN p_Image BLOB)
4

⊖ BEGIN

5
6
           Update shoes
7
           SET Name = COALESCE(p_Name, Name),
               Brand = COALESCE(p_Brand, Brand),
               Size = COALESCE(p_Size, Size),
9
               Color = COALESCE(p_Color, Color),
10
               Price = COALESCE(p_Price, Price),
11
               Image = COALESCE(p_Image, Image)
12
           WHERE shoes ID = p shoes ID;
13
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 1	_	_		25.55	PI OP

	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 //
1
2 • ⊖ CREATE PROCEDURE update_staff(IN p_staff_ID INT, IN p_ST_FirstName VARCHAR(45),
3
                                   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))
    \ominus BEGIN
5
6
          Update staff
7
           SET ST_FirstName = COALESCE(p_ST_FirstName, ST_FirstName),
               ST_LastName = COALESCE(p_ST_LastName, ST_LastName),
8
               ST_Address = COALESCE(p_ST_Address,ST_Address),
9
               ST_Email = COALESCE(p_ST_Email,ST_Email),
10
               ST_DoB = COALESCE(p_ST_DoB,ST_DoB),
11
               ST_Password = COALESCE(p_ST_Password, ST_Password)
12
13
           WHERE staff_ID = p_staff_ID;
14
       END
       //
```

## 1 • 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
	staff_ID	Charles	Baker	15648 Bend Street	CBaker@gmail.com	1981-01-19	Password
	staff_ID 1 2	Charles Heather	Baker Wonder	15648 Bend Street 12348 Yule Drive	CBaker@gmail.com WonderWoman@gmail.com	1981-01-19 1993-04-12	Password Password
*	staff_ID	Charles	Baker	15648 Bend Street	CBaker@gmail.com	1981-01-19	Password
*	staff_ID 1 2	Charles Heather	Baker Wonder	15648 Bend Street 12348 Yule Drive	CBaker@gmail.com WonderWoman@gmail.com	1981-01-19 1993-04-12	Password Password
*	staff_ID  1  2  3	Charles Heather David	Baker Wonder Brash	15648 Bend Street 12348 Yule Drive 75479 Canyon Oaks	CBaker@gmail.com WonderWoman@gmail.com BigBrash@yahoo.com	1981-01-19 1993-04-12 1991-12-30	Password Password Password
*	staff_ID  1  2  3  4	Charles Heather David Monica	Baker Wonder Brash Liu	15648 Bend Street 12348 Yule Drive 75479 Canyon Oaks 13993 Benard Avenue	CBaker @gmail.com WonderWoman @gmail.com BigBrash @yahoo.com TheM@gmail.com	1981-01-19 1993-04-12 1991-12-30 2003-06-24	Password Password Password Password
*	staff_ID 1 2 3 4 5	Charles Heather David Monica Dion	Baker Wonder Brash Liu Tempest	15648 Bend Street 12348 Yule Drive 75479 Canyon Oaks 13993 Benard Avenue 11404 State Court	CBaker @gmail.com WonderWoman @gmail.com BigBrash @yahoo.com TheM@gmail.com StormingD @yahoo.com	1981-01-19 1993-04-12 1991-12-30 2003-06-24 1989-02-13	Password Password Password Password Password
* •	staff_ID 1 2 3 4 5 6	Charles Heather David Monica Dion Emily	Baker Wonder Brash Liu Tempest Vega	15648 Bend Street 12348 Yule Drive 75479 Canyon Oaks 13993 Benard Avenue 11404 State Court 361 Maple Oak Drive	CBaker @gmail.com Wonder Woman @gmail.com BigBrash @yahoo.com The M@gmail.com Storming D @yahoo.com Emily Vega @yahoo.com	1981-01-19 1993-04-12 1991-12-30 2003-06-24 1989-02-13 2001-03-10	Password Password Password Password Password Password Password
	staff_ID  1  2  3  4  5  6  7	Charles Heather David Monica Dion Emily Momo	Baker Wonder Brash Liu Tempest Vega Harb	15648 Bend Street 12348 Yule Drive 75479 Canyon Oaks 13993 Benard Avenue 11404 State Court 361 Maple Oak Drive 12348 Cinnamon Street	CBaker @gmail.com WonderWoman @gmail.com BigBrash @yahoo.com TheM@gmail.com StormingD @yahoo.com EmilyVega @yahoo.com HarbBro @gmail.com	1981-01-19 1993-04-12 1991-12-30 2003-06-24 1989-02-13 2001-03-10 1987-11-15	Password Password Password Password Password Password Password Password
*	staff_ID  1  2  3  4  5  6  7  8	Charles Heather David Monica Dion Emily Momo Kasto	Baker Wonder Brash Liu Tempest Vega Harb Issa	15648 Bend Street 12348 Yule Drive 75479 Canyon Oaks 13993 Benard Avenue 11404 State Court 361 Maple Oak Drive 12348 Cinnamon Street 7845 Ember Creek	CBaker@gmail.com WonderWoman@gmail.com BigBrash@yahoo.com TheM@gmail.com StormingD@yahoo.com EmilyVega@yahoo.com HarbBro@gmail.com RollingIssa@yahoo.com	1981-01-19 1993-04-12 1991-12-30 2003-06-24 1989-02-13 2001-03-10 1987-11-15 1989-05-07	Password Password Password Password Password Password Password Password Password

### viewcustomers()

10

Livia

Yusuf

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

```
DELIMITER //
                     2 •
                             CREATE PROCEDURE viewcustomers()
                     3

→ BEGIN

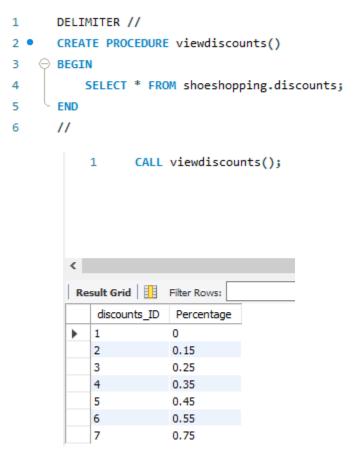
                                  SELECT * FROM shoeshopping.customers;
                     4
                             END
                     5
                             //
                     6
  1
          CALL viewcustomers();
Result Grid Filter Rows:
                                             Export: Wrap Cell Content: IA
   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 Cyde Cirde
                                                                 FranzTheGreat@gmail.com
                                                                                           1981-03-14
                                        12358 Enduring Path
   3
            Ruben
                          Thanada
                                                                 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
                                        81131 Market Avenue
                                                                 EGLLC@gmail.com
                                                                                           1990-05-23
            Erato
                          Gupta
   7
            Jasos
                          Khalil
                                        4587 Lincoln Drive
                                                                 BestJasos@gmail.com
                                                                                           1984-01-04
   8
            Edgar
                          Idwal
                                    Khalil 22 Open Forest
                                                                 MoonShoe@yahoo.com
                                                                                           1988-03-22
   9
                                        77988 Column Court
                                                                 GiGi123@gmail.com
                          Severina
                                                                                           1974-12-02
```

15742 Sinking Sea

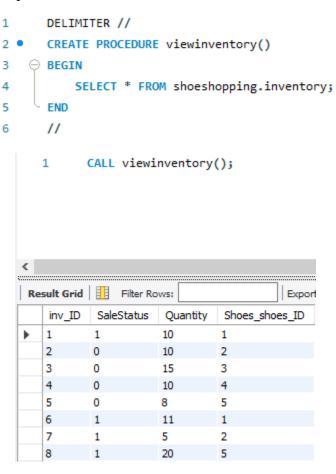
MsLiv@yahoo.com

1974-08-14

## • viewdiscounts()



• viewinventory()



## • vieworders()

20

2020-11-14 09:50:12 0

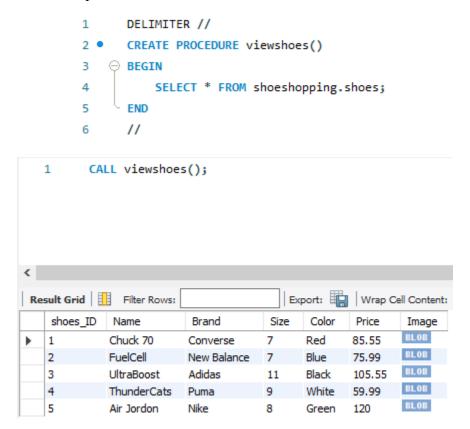
175.15

```
DELIMITER //
                              1
                              2 •
                                       CREATE PROCEDURE vieworders()
                              3

→ BEGIN

                                            SELECT * FROM shoeshopping.orders;
                              4
                              5
                                       END
                                       //
                              6
          CALL vieworders();
Result Grid Filter Rows:
                                         Export: Wrap Cell Content: 🔼
    order_ID OrderDate
                                            TotalAmount Inventory_inv_ID
                                                                          staff_staff_ID customers_cust_ID
                                 OrderStatus
                                                                                                         discounts_discounts_ID
             2019-10-29 14:56:59
                                            300.55
             2019-11-12 10:32:10
                                                                         2
   2
                                0
                                            105.55
                                                        2
                                                                                       5
                                                                                                        0
                                            540.45
                                                                                                        0
   3
             2019-11-24 13:01:45
                                                         3
                                                                                       8
                                0
   4
             2019-12-08 07:43:22 0
                                            360
                                                        1
                                                                         3
                                                                                                        1
             2019-12-02 09:13:55 0
                                            75.55
   5
             2019-12-11 18:30:21 0
                                            1242.95
                                                        5
                                                                         7
                                                                                                        0
   6
             2019-12-21 12:06:57 0
                                            212.56
                                                                         6
                                                                                       6
   7
                                                                         8
                                                                                                        0
             2019-12-24 09:40:13 0
                                                        2
                                                                                       7
                                            90.96
   8
             2019-12-24 15:23:07 0
                                            173.45
                                                                         3
                                                         5
                                                                                       5
   9
             2020-01-13 09:51:39 0
                                            75.33
                                                                         1
                                                                                       10
                                                                                                        0
   10
             2020-01-25 20:42:19
                                            966.41
                                                         1
                                                                         10
   11
             2020-01-18 06:50:07 1
                                            1475.45
                                                                         5
                                                                                                        3
             2020-02-04 10:48:32 1
                                            85.25
                                                                         3
                                                                                       6
   12
             2020-02-17 13:12:44 1
                                            613.99
                                                                         8
   13
                                                                         9
                                                                                                        2
   14
             2020-03-12 09:19:47 1
                                            219.96
                                                                                       2
   15
             2020-04-09 04:23:01 1
                                            96.75
                                                                         2
   16
             2020-05-02 19:59:28 0
                                            1313.36
                                                         5
                                                                         7
                                                                                       10
                                                                                                        1
             2020-05-23 04:06:52 1
                                                                         4
                                                                                                        3
                                                        3
                                                                                       4
   17
                                            154.15
             2020-07-06 11:20:44 0
   18
                                            590
                                                         2
                                                                         6
                                                                                       9
                                                                                                        2
   19
             2020-08-20 14:12:05 0
                                            305.55
                                                        5
                                                                         1
                                                                                       5
                                                                                                        1
```

## viewshoes()



• viewstaff()

```
DELIMITER //
                            2 •
                                     CREATE PROCEDURE viewstaff()
                            3

→ BEGIN

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

# **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.

```
1
       DELIMITER //
 2
 3 •
       CREATE TRIGGER after_shoes_insert
 4
       AFTER INSERT
 5
       ON shoes FOR EACH ROW
 6

→ BEGIN

           INSERT INTO inventory(SaleStatus, Quantity, Shoes_shoes_ID)
7
8
           VALUES (0, 5, NEW.shoes_ID);
     L END
9
10
       //
```