CS6360.001 DATABASE DESIGN

PROJECT: DATABASE SYSTEM FOR UBER EATS

TEAM NUMBER: Uber Eats – T5

TEAM MEMBERS

Harsha Priya Daggubati (HXD220007)

Surya Teja Sri Jonnada (STJ220000)

<u>Step 7:</u> PL/SQL: Define two relevant stored procedures and two triggers (they should have a meaningful application in real-world cases).

1. Generate coupons and add them to the customer account if he or she is a frequent user of the app, based on the criteria that number of orders in 2 consecutive months are greater than 5. If number of orders in a year are at least 50, then make them an uber one member. This action is triggered when user tries to place an order in the app, if the requirements match, then coupon codes are added to the account.

```
delimiter //

use uberEatsDatabase;

CREATE PROCEDURE generateCoupon(IN customerId CHAR(10))

BEGIN

        DECLARE total_orders INT;

        DECLARE total_orders_year INT;

        SELECT COUNT(orderId) INTO total_orders from ORDER WHERE customer_id =
customerId AND MONTH(order_date) - MONTH(SYSDATE()) <= 2;

        SELECT COUNT(orderId) INTO total_orders_year from ORDER WHERE customer_id =
customerId AND YEAR(order_date) - YEAR(SYSDATE()) <= 1;

        IF total_orders > 6 THEN

                DECLARE couponId CHAR(10)

                SELECT ‘UBRETS’ + CAST(FLOOR(RAND() * (10000-100) + 100) as varchar(4)) INTO
couponId;

                INSERT INTO DISCOUNTCOUPON(couponCode, discount_percentage,
min_order_amount) VALUES(couponId, 0.25, 20);

                INSERT INTO GETS(customerId, couponCode) VALUES(customerId, couponId);

        IF total_orders_year > 50 THEN

        ALTER TABLE CUSTOMER
```

```
                SET isUberOneMember = 1
        WHERE CUSTOMER.customer_id = customerId;
END //


CREATE TRIGGER addCouponsToAccount BEFORE INSERT ON ORDER
        CALL checkAndGenerateCoupon(NEW.customerId);
```

2. Update the average rating of delivery partner after user rates the delivery made by a delivery partner after a while on the app. The dollar share per order of the delivery partner should also be updated based on the rating.

```
delimiter //
use uberEatsDatabase;
CREATE PROCEDURE calculateAverage(IN id CHAR(10), OUT rating DOUBLE)
BEGIN
 SELECT AVG(rating) INTO rating from delivers GROUP BY delivery_partner_id
        AND delivery_partner_id = id;
END


CREATE TRIGGER updateRating AFTER UPDATE ON DELIVERS
        DECLARE avgRating INT;
        IF NEW.rating > 0 THEN
        CALL calculateAverage(NEW.delivery_partner_id, avgRating);


        DECLARE dollarShare DOUBLE;
        IF avgRating < 1 THEN
```

```
                SET dollarShare = 0.05;
        ELSEIF avgRating < 2 THEN
                SET dollarShare = 0.1;
        ELSEIF avgRating < 3 THEN
                SET dollarShare = 0.15;
        ELSEIF avgRating < 4 THEN
                SET dollarShare = 0.25;
        ELSEIF avgRating < 5 THEN
                SET dollarShare = 0.3;
        ELSE
                SET dollarShare = 0.35;

ALTER TABLE DeliveryPartner
        SET rating = avgRating AND
        $_Share_order = dollarShare
        WHERE delivery_partner_id = NEW.delivery_partner_id;
```