Procedure which does group by information

```
CREATE OR REPLACE PROCEDURE get_product_category_summary
IS
BEGIN
  FOR c IN (SELECT Sneaker_Model.brand, COUNT(*) AS ProductCount,
SUM(Sneaker_Model_Variant.quantity_available) AS TotalQuantity, AVG(Sneaker_Model.price) AS AveragePrice
            FROM Sneaker_Model
            JOIN Sneaker_Model_Variant ON Sneaker_Model.sneaker_model_id = Sneaker_Model_Variant.sneaker_model_id
            GROUP BY Sneaker_Model.brand)
  LOOP
    DBMS_OUTPUT.PUT_LINE(c.brand || ': ' ||  c.ProductCount || ' products, ' || c.TotalQuantity || ' total quantity, average
price: '||  c.AveragePrice);
  END LOOP;
END;
```

```
begin
    get_product_category_summary;
end;
```

Function which counts the number of records

```
CREATE OR REPLACE FUNCTION count_records(Buyer IN VARCHAR) RETURN NUMBER AS
  v_count NUMBER;
BEGIN
  EXECUTE IMMEDIATE 'SELECT COUNT(*) FROM ' || Buyer INTO v_count;
  RETURN v_count;
END;

DECLARE
  v_count NUMBER;
BEGIN
  v_count := count_records('Buyer');
  DBMS_OUTPUT.PUT_LINE('Number of records: ' ||  v_count);
END;
```

Procedure which uses SQL%ROWCOUNT to determine the number of rows affected

```
CREATE OR REPLACE PROCEDURE update_price(price_change IN DECIMAL) AS
BEGIN
  UPDATE Sneaker_Model
  SET price = price + price_change;

  IF SQL%ROWCOUNT = 0 THEN
    DBMS_OUTPUT.PUT_LINE('No rows were updated.');
  ELSE
    DBMS_OUTPUT.PUT_LINE(SQL%ROWCOUNT || ' rows were updated.');
  END IF;
END;


BEGIN
  update_price(10000.00); -- passing a price_change value of 10000.00
END;
```

Create a trigger before insert on any entity which will show the current number of rows in the table

```
CREATE OR REPLACE TRIGGER show_row_count
BEFORE INSERT ON Courier
DECLARE
  v_row_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_row_count FROM Buyer;
  DBMS_OUTPUT.PUT_LINE('The current number of rows in the table is: ' || v_row_count);
END;
```

Add user-defined exception which disallows to enter title of item (e.g. book) to be less than 5 characters

```
CREATE OR REPLACE PROCEDURE insert_item (
  c_courier_id IN NUMBER,
  c_courier_name IN VARCHAR2,
  c_courier_phone_number IN VARCHAR2,
  c_courier_email IN VARCHAR2
) AS
  title_length EXCEPTION;
BEGIN
  IF LENGTH(c_courier_phone_number) < 5 THEN
    RAISE title_length;
  END IF;

  INSERT INTO Courier (courier_id, courier_name,courier_phone_number,courier_email)
  VALUES (c_courier_id, c_courier_name,c_courier_phone_number,c_courier_email);

  COMMIT;
EXCEPTION
  WHEN title_length THEN
    RAISE_APPLICATION_ERROR(-20001, 'Courier phone number must be at least 5 characters long');
END;

BEGIN
    insert_item(14,'Mukan', '11111', 'as@mail.xyz');
END;
```