“TECHNODOM”

1)ADILBECK YERNUR

2)ISKENDER DARIGA

3)ALITURLIYEV AIDYN

4)KALDANBEKOV BAKDAULET

INTRODUCTION:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

In the modern world, technology has become an integral part of our daily lives, and databases play an important role in their functioning. One of the most popular and widespread technological sectors is the technodome, which is a comprehensive system for automating the management of construction and operation of buildings and structures.

Technodom uses databases to store and process information about buildings, equipment, design solutions and other aspects of construction and operation of facilities. Technodom databases allow you to effectively manage projects and provide control over all stages of construction and operation of buildings and structures.

The goal of our project is to create a database that would improve and optimize the management of the construction and operation of buildings and structures. In this report, we will look at the basic principles and technologies that underlie the creation of a database for a technodom and describe the process of its development and implementation.

Thus, this project is of great importance for improving the efficiency and quality of work in the construction industry, and is one of the key tools for increasing the competitiveness of companies working in this field.

Normalization\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1)Customer:

1NF-The table is in the 1NF because there are no duplicate rows and it is atomic

2NF-The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

3NF-The table is in the 3NF because there is no functional dependency

2)ITEMS:

1NF- The table is in the 1NF because there are no duplicate rows and it is atomic

2NF- The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

3NF- The table is in the 3NF because there is no functional dependency

3)BASKET:

1NF- The table is in the 1NF because there are no duplicate rows and it is atomic

2NF- The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

3NF- The table is in the 3NF because there is no functional dependency

4)PAYMENT:

1NF- The table is in the 1NF because there are no duplicate rows and it is atomic

2NF- The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

3NF- The table is in the 3NF because there is no functional dependency

5)CARD:

1NF- The table is in the 1NF because there are no duplicate rows and it is atomic

2NF- The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

3NF- The table is in the 3NF because there is no functional dependency

6)RESPONSE:

1NF- The table is in the 1NF because there are no duplicate rows and it is atomic

2NF- The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

3NF- The table is in the 3NF because there is no functional dependency

7)DELIVERY:

1NF- The table is in the 1NF because there are no duplicate rows and it is atomic

2NF- The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

3NF- The table is in the 3NF because there is no functional dependency

8)ADMIN:

1NF- The table is in the 1NF because there are no duplicate rows and it is atomic

2NF- The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

3NF- The table is in the 3NF because there is no functional dependency

9)POPULAR\_ITEMS:

1NF- The table is in the 1NF because there are no duplicate rows and it is atomic

2NF- The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

3NF- The table is in the 3NF because there is no functional dependency

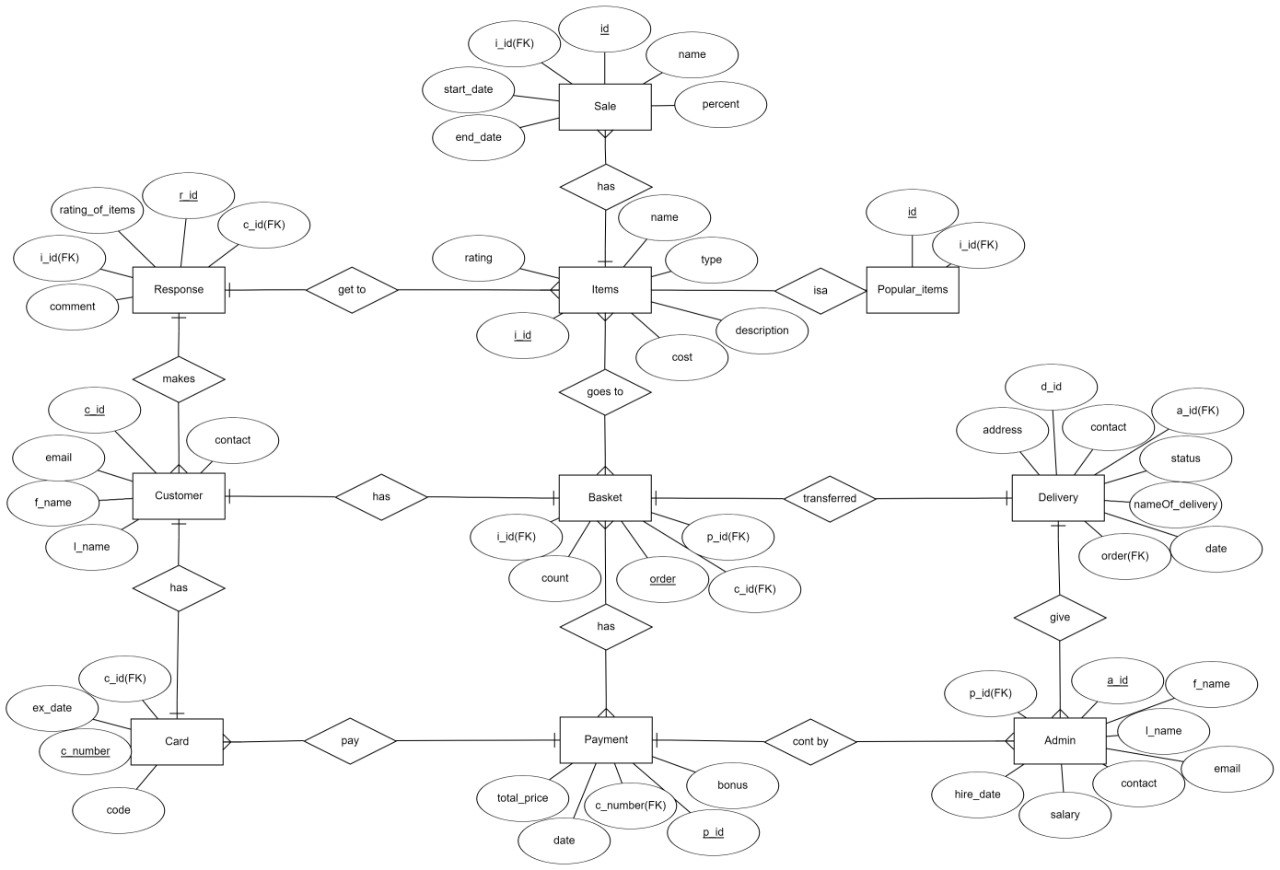
10)SALE:

1NF- The table is in the 1NF because there are no duplicate rows and it is atomic

2NF- The table is in the 2NF because every column that is not a primary key is completely dependent on the primary key.

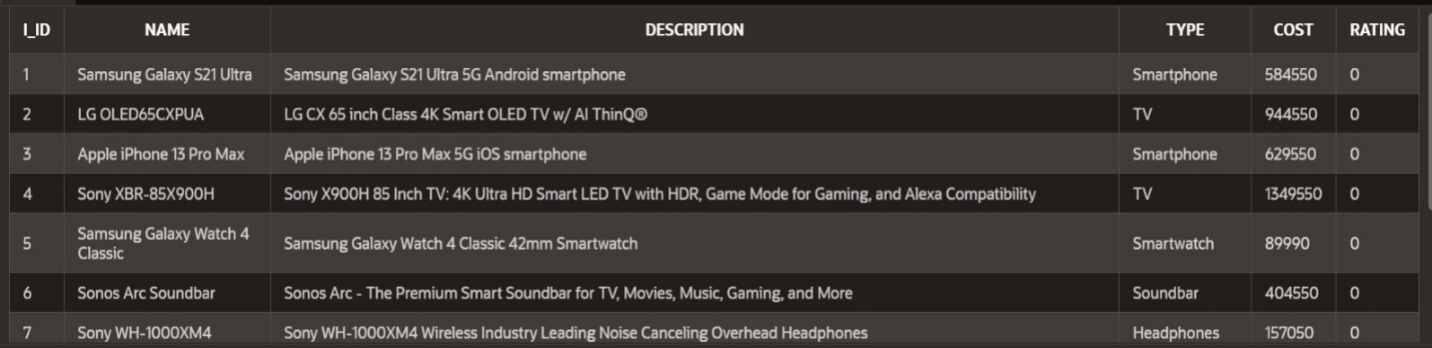
3NF- The table is in the 3NF because there is no functional dependency

ER DIAGRAM:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

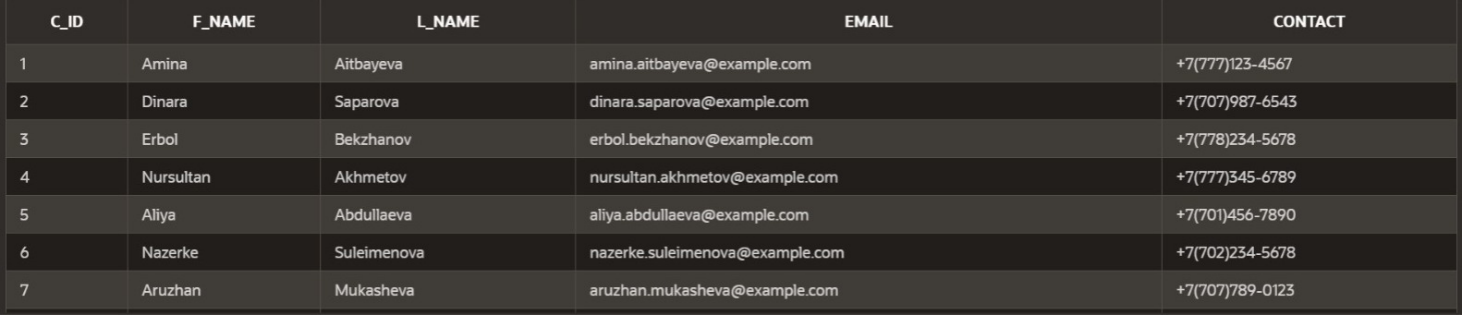


TABLES:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

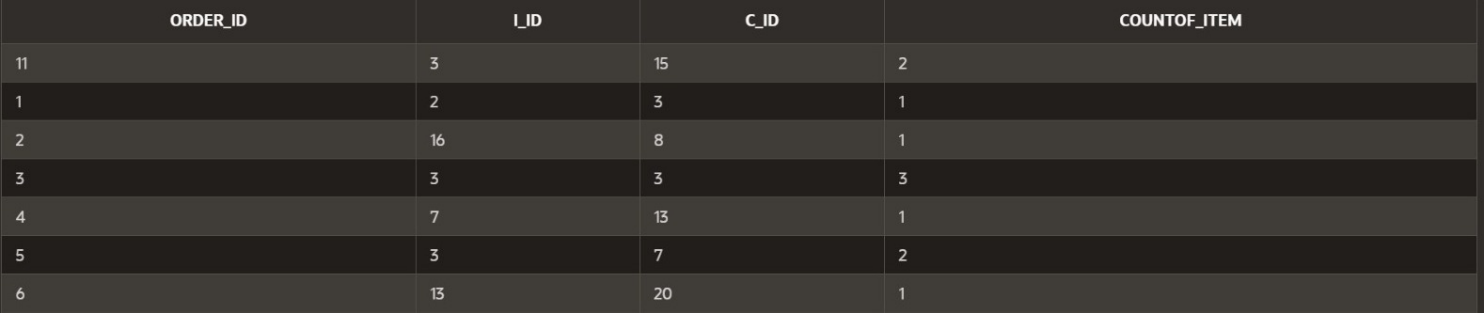
1)ITEMS: THIS TABLE SHOWS INFORMATION ABOUT THE PRODUCTS



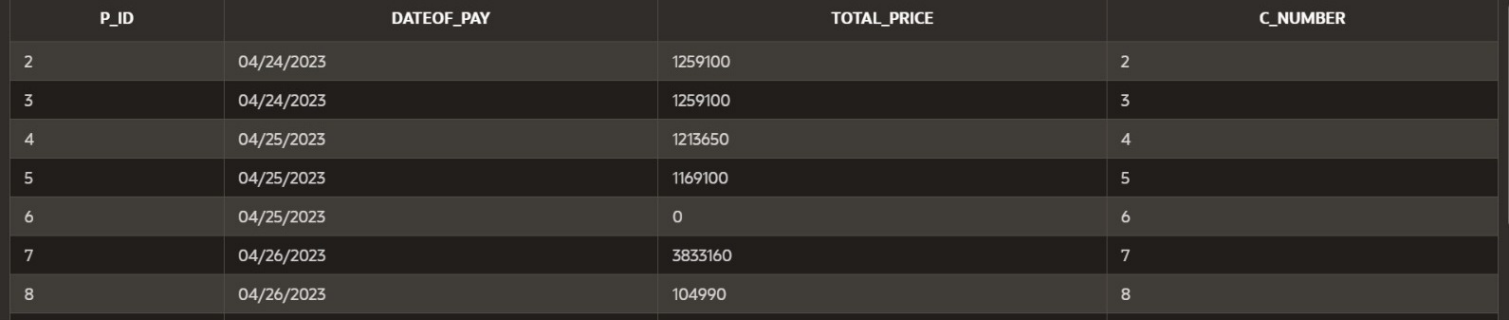
2)CUSTOMER: THIS TABLE SHOWS INFORMATION ABOUT CUSTOMERS



3)BASKET: THIS TABLE SHOWS WHAT IS INSIDE IN THE BASKETS



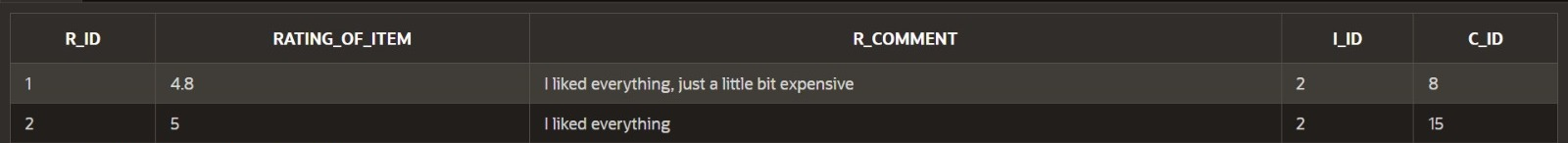
4)PAYMENT: THIS TABLE SHOWS INFORMATION ABOUT PAYMENTS



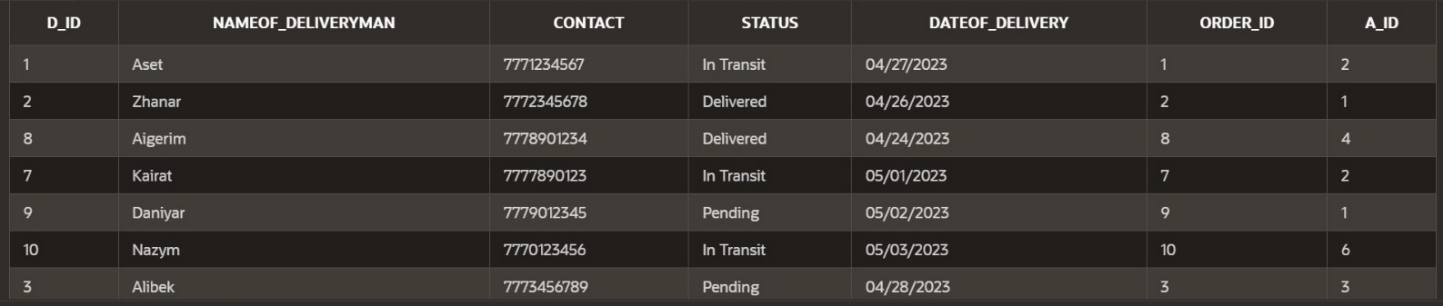
5)CARD: THIS TABLE SHOWS INFORMATION ABOUT THE CARDS



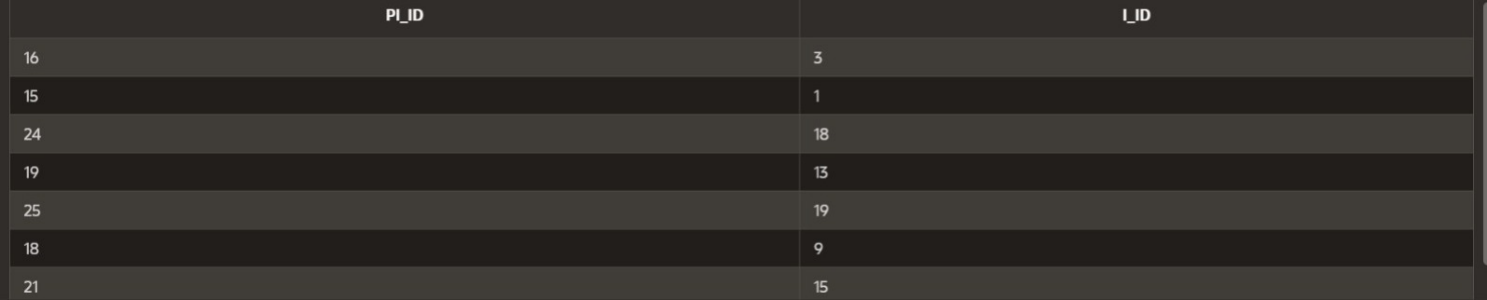
6)RESPONSE: THIS TABLE SHOWS PRODUCT REVIEWS ABOUT STORES



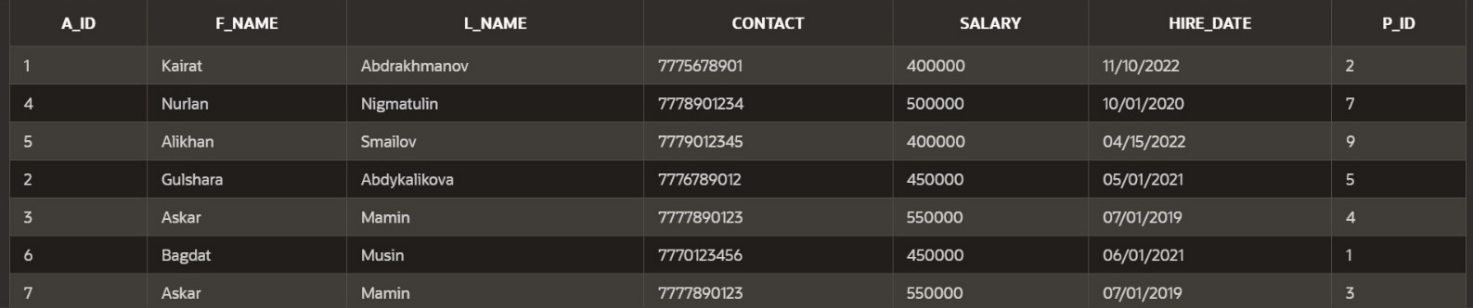
7)DELIVERY: THIS TABLE SHOWS INFORMATION ABOUT DELIVERIES



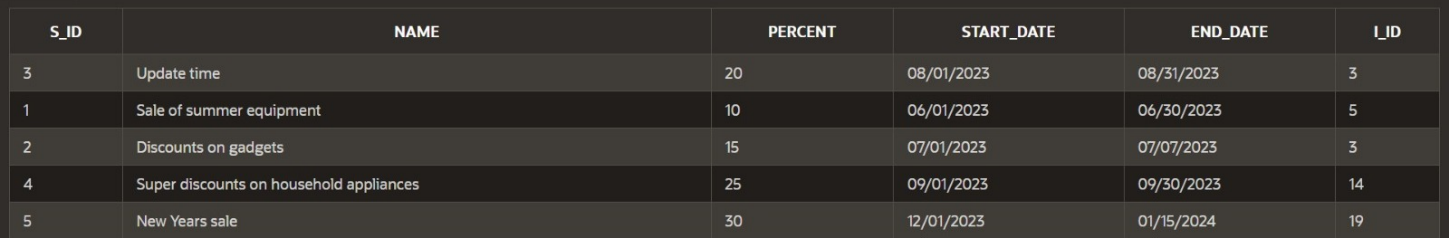
8)POPULAR\_ITEMS: THIS TABLE SHOWS THE INFORMATION OF A POPULAR PRODUCT AMONG BUYERS



9)ADMIN: THIS TABLE SHOWS INFORMATION ABOUT ADMINS



10)SALE: THIS TABLE SHOWS INFORMATION ABOUT PROMOTIONS IN THE STORE



PROCEDURES:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**1)**[**procedure which goes group by information**](javascript:sc_postSavedSQL('4017207659473940252','procedure%20which%20goes%20group%20by%20information','','SC','WKSP_AIDYN');)**:**

CREATE OR REPLACE PROCEDURE calculate\_avg\_cost IS

BEGIN

FOR i IN (SELECT type, AVG(cost) AS avg\_cost FROM items GROUP BY type)

LOOP

DBMS\_OUTPUT.PUT\_LINE('Average cost for ' i.type ': ' || i.avg\_cost);

END LOOP;

END;

BEGIN

calculate\_avg\_cost;

END;

**2)** [**Procedure which uses SQL%ROWCOUNT to determine the number of rows affected**](javascript:sc_postSavedSQL('4038416113452359536','Procedure%20which%20uses%20SQL\u0025ROWCOUNT%20to%20determine%20the%20number%20of%20rows%20affected','','SC','WKSP_AIDYN');)**:**

**CREATE OR REPLACE PROCEDURE update\_item\_cost(item\_id IN NUMBER, new\_cost IN NUMBER) IS**

**BEGIN**

**UPDATE Items SET cost = new\_cost WHERE i\_id = item\_id;**

**IF SQL%ROWCOUNT = 0 THEN**

**DBMS\_OUTPUT.PUT\_LINE('No item found with ID ' || item\_id);**

**ELSE**

**DBMS\_OUTPUT.PUT\_LINE('The cost of item ' || item\_id || ' has been updated.');**

**END IF;**

**END;**

**BEGIN**

**update\_item\_cost(1, 585000);**

**END;**

EXSEPTION:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**1)**[**exception that dissalows to insert card with expired date(bonus)**](javascript:sc_postSavedSQL('4061191278985588939','exception%20that%20dissalows%20to%20insert%20card%20with%20expired%20date(bonus)','','SC','WKSP_AIDYN');)**:**

**CREATE OR REPLACE TRIGGER insert\_card\_expDate\_exc**

**BEFORE INSERT ON card**

**FOR EACH ROW**

**DECLARE**

**current\_date DATE;**

**expired\_date exception;**

**BEGIN**

**current\_date := SYSDATE;**

**IF :NEW.ex\_date < current\_date THEN**

**raise expired\_date;**

**END IF;**

**EXCEPTION**

**WHEN expired\_date THEN**

**RAISE\_APPLICATION\_ERROR(-20003, 'Your card is expired');**

**END;**

**insert into card(c\_number, ex\_date, code, c\_id) values (11, '02/11/2022', 372, 2)**

**2)** [**--Add user-defined exception which disallows to enter title of item (e.g. book) to be less than 5 characters:**](javascript:sc_postSavedSQL('4055710866554364621','--Add%20user-defined%20exception%20which%20disallows%20to%20enter%20title%20of%20item%20(e.g.%20book)%20to%20be%20less%20than%205%20characters:','','SC','WKSP_AIDYN');)

**CREATE OR REPLACE TRIGGER insert\_item\_name\_exc**

**BEFORE INSERT ON items**

**FOR EACH ROW**

**DECLARE**

**name\_length NUMBER;**

**name\_length\_exception EXCEPTION;**

**BEGIN**

**name\_length := LENGTH(:NEW.name);**

**IF name\_length < 5 THEN**

**raise name\_length\_exception;**

**END IF;**

**EXCEPTION**

**WHEN name\_length\_exception THEN**

**RAISE\_APPLICATION\_ERROR(-20001, 'Item name must be minimum 5 characters');**

**END;**

**insert into items(i\_id, name, description, type, cost, rating) values(50, 'asdf', 'sfdf', 'sfs', 224145, 4)**

**3)** [**exception that checks for validation email of customer(bonus)**](javascript:sc_postSavedSQL('4058966009706998663','exception%20that%20checks%20for%20validation%20email%20of%20customer(bonus)','','SC','WKSP_AIDYN');)**:**

**CREATE OR REPLACE TRIGGER insert\_customer\_email\_exc**

**BEFORE INSERT ON customer**

**FOR EACH ROW**

**DECLARE**

**email\_sym NUMBER;**

**check\_email\_symbol exception;**

**BEGIN**

**email\_sym := INSTR(:NEW.email, '@');**

**IF email\_sym = 0 THEN**

**raise check\_email\_symbol;**

**END IF;**

**EXCEPTION**

**WHEN check\_email\_symbol THEN**

**RAISE\_APPLICATION\_ERROR(-20009, 'Email must contain "@" symbol');**

**DBMS\_OUTPUT.PUT\_LINE('Email must contain @ symbol');**

**END;**

**insert into customer(c\_id, f\_name, l\_name, email, contact) values (35, 'asfd', 'asdf', 'sdfas.com', 25324534)**

TRIGGERS:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**1)create or replace trigger basket\_trig**

**before INSERT on Basket**

**for each row**

**begin**

**Update Payment**

**set total\_price = total\_price + (select sum(cost \* :New.countOf\_item) from Items where i\_id = :New.i\_id)**

**where c\_number = (select c\_number from Card where c\_id = :New.c\_id); end;**

**2)create or replace trigger**

**delete\_basket\_trig**

**after Delete on Basket**

**for each row**

**begin**

**Update Payment set total\_price = total\_price - (select sum(cost \* :Old.countOf\_item) from Items where i\_id = :Old.i\_id)**

**where c\_number = (select c\_number from Card where c\_id = :Old.c\_id); end;**

**3)CREATE OR REPLACE TRIGGER**

**rating\_trig**

**BEFORE INSERT ON Response**

**FOR EACH ROW**

**BEGIN**

**UPDATE Items SET rating = (SELECT AVG(rating\_of\_item) FROM Response Group by i\_id) WHERE i\_id = :new.i\_id; END;**

**4)** [**Create a trigger before insert on any entity which will show the current number of rows in the table:**](javascript:sc_postSavedSQL('4052700861586753687','Create%20a%20trigger%20before%20insert%20on%20any%20entity%20which%20will%20show%20the%20current%20number%20of%20rows%20in%20the%20table:','','SC','WKSP_AIDYN');)

**CREATE OR REPLACE TRIGGER display\_row\_count**

**BEFORE INSERT ON items**

**FOR EACH ROW**

**DECLARE**

**rows NUMBER;**

**BEGIN**

**SELECT COUNT(\*) INTO rows FROM items;**

**DBMS\_OUTPUT.PUT\_LINE('Current row count: ' || rows);**

**END;**

**insert into items(i\_id, name, description, type, cost, rating) values (43, 'safd', 'agfgf', 'sdaf', 325323, 3)**

FUNCTION:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. [**function that counts number of responsed customers(bonus)**](javascript:sc_postSavedSQL('4043056186206442692','function%20that%20counts%20number%20of%20responsed%20customers(bonus)','','SC','WKSP_AIDYN');)**:**

**CREATE OR REPLACE FUNCTION number\_responsed\_customers (**

**item\_id IN NUMBER,**

**customer\_id IN NUMBER**

**)**

**RETURN NUMBER**

**IS**

**n NUMBER;**

**BEGIN**

**SELECT COUNT(\*)**

**INTO n**

**FROM response r**

**JOIN items i ON r.i\_id = i.i\_id**

**WHERE i.i\_id = item\_id**

**AND r.c\_id = customer\_id;**

**RETURN n;**

**END;**

**DECLARE**

**n NUMBER;**

**BEGIN**

**n := number\_responsed\_customers(2, 8);**

**DBMS\_OUTPUT.PUT\_LINE('The customer has left ' || n || ' response(-s) for item 2.');**

**END;**

**2)**[**function which counts the number of records**](javascript:sc_postSavedSQL('4017062325090457146','function%20which%20counts%20the%20number%20of%20records','','SC','WKSP_AIDYN');)**:**

**CREATE OR REPLACE FUNCTION Number\_of\_Delivered RETURN NUMBER**

**IS**

**delivered\_count NUMBER;**

**BEGIN**

**SELECT COUNT(\*)**

**INTO delivered\_count**

**FROM delivery**

**WHERE status = 'Delivered';**

**RETURN delivered\_count;**

**END;**

**DECLARE**

**n NUMBER;**

**BEGIN**

**n := Number\_of\_Delivered();**

**DBMS\_OUTPUT.PUT\_LINE('The number of deliveries with status delivered is: ' || n);**

**END;**

GITHUB URL:\_\_ <https://github.com/johenADN/database-midterm>