**UNIVERSITY OF GHANA**

**Department of Computer Engineering**

**SCHOOL OF ENGINEERING SCIENCES**

**COLLEGE OF BASIC AND APPLIED SCIENCES**

**FIRST SEMESTER 2022/2023 ACADEMIC YEAR**

**COURSE CODE: CPEN 211 DATABASE SYSTEMS.**

**COURSE INSTRUCTOR: Assiamah Korankye John**

**Name: DOE AGUDEY DANIEL**

**ID:10956661**

**PROJECT TWO(2)**

**DATE:30/04/2023**

## QUESTION 1

## HOW I WILL GO ABOUT THE PROBLEM

### Problem identification and understanding.

As a database consultant, I would use a systematic approach to make the Airport Shell fuel station run automatically. First, I would find out how things are currently done at the gas station by looking at the manual processes and tasks involved. I will examine all the different steps and tools involved. I will talk to the people who are interested in our work, like the boss, workers, and customers, to make sure that I understand the problem very well.

### Information gathering

After understanding all the processes, I would figure out what the new automated system needs. This will be done by gathering information from bosses, customers, workers and all sources of information that will be of great help .This would help me make a system that keeps track of a lot of information in an organized way. From gathering and analyzing the information it may become apparent that the database must contain relations for customers, types of fuel, tanks and pumps for fuel, people who work at the pump, a record of sales, and methods of payment.

### Design a blueprint for the database(ER diagram)

After enough information has been gathered and analyzed ,I will draw a conceptual diagram known as the ER diagram to show what information our database will store, using shapes that represent the entities we want to store and lines to show how they are connected. The ER diagram shows how things are related to each other and what they are made up of. For instance, an entity called "Sales Transactions" is related to another entity called "Customers." , whenever a the customer makes a transaction by purchasing fuel .After making the ER diagram, I will construct the database schema that will be required for the structure of the tables in the database.

### Design the database

I will design and build a database management system based on the ER diagram and the schema which includes all the necessary entities, attributes, and relationships. Then I will develop a user interface for the system to enable easy data entry and retrieval by authorized personnels.

### Testing and experimentation

Once I finish designing how the database looks and functions for each user, I will make sure it works correctly. This means checking if the database can handle a lot of information, is strong, and can expand if needed, finally I will also run multiple queries on the database. I will check if all functions work correctly, and make sure that the way things are related to each other is done the right way in the database.

**Documentation of how the database functions**

After making sure that everything works properly, I would write down how the database is organized, including what tables it has and how they are connected. This document will help the fuel station management and workers and any future programmers who might work with the database system in the future have an easy way of navigating through the system .

**Communication and education**

After the whole documentation of the database schema, including the ER diagram, table structures, and relationships of the designed database. I will provide training and support to the filling station staff members on how to use the system effectively if the need arises.

**Conclusion**

To make the Airport Shell filling station easier to use and automatic, I figured out what is currently manually and came up with a plan for how computers can do it instead, by building a highly advanced database system. The database program helps to keep track of everything and make it easy to handle large processes. Finally, I tested and educated the workers about how the database functions.

**QUESTION 2**

**Entities**

- Fuel inventory
- Fuel tank
- Suppliers
- Employees
- Customers
- Transactions

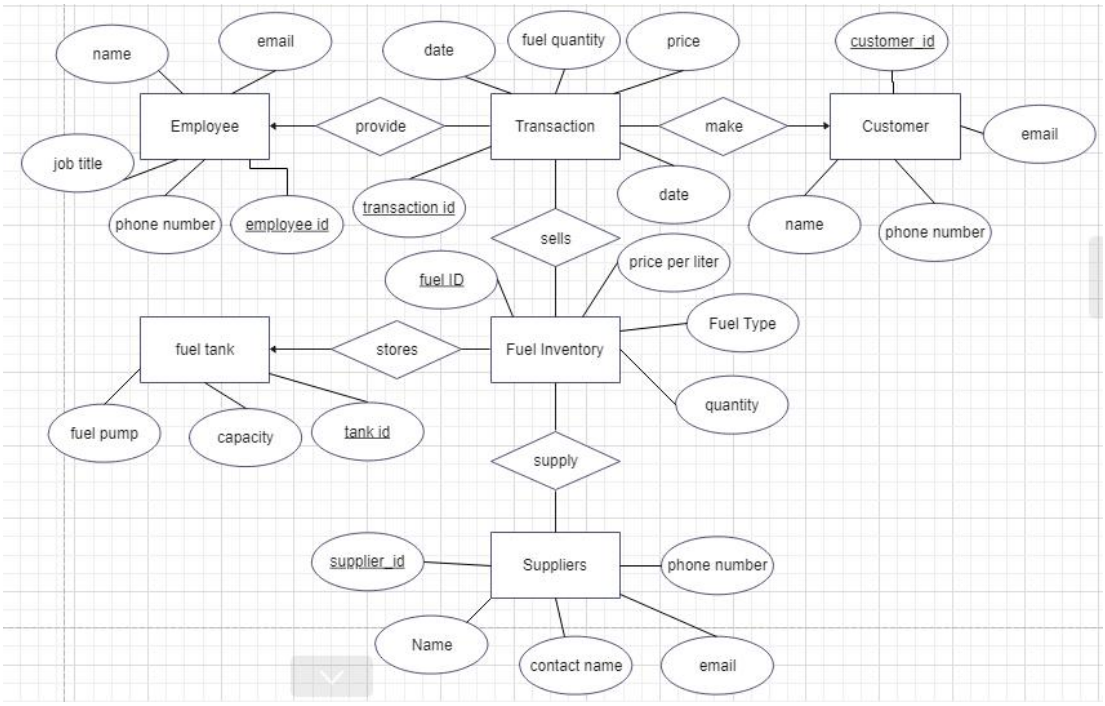
**Relationship between Entities**

| **Entity** | **Relationship** | **Entity** |
|---|---|---|
| Employee | Provides | Transaction |
| Customer | Makes | Transaction |
| Transaction | Sells | Fuel inventory |
| Fuel tank | Stores | Fuel inventory |
| Suppliers | Supply | Fuel inventory |

1. Fuel tanks and supplier have a one to many relationship, where a fuel tank can be **supplied** by multiple suppliers, and a supplier can **supply** fuel to multiple fuel tanks.
2. Fuel tanks and fuel inventory have a one,to,many relationship, where a fuel tank can have multiple types of fuel, and a type of fuel can only be stored in one fuel tank.
3. Supplier and fuel inventory have a one to many relationship, where a supplier can supply multiple types of fuel, and a type of fuel can be supplied by multiple suppliers.
4. Customer and sale transaction have a one to many relationship, where a customer can make multiple transactions, and a transaction can only be associated with one customer.
5. Fuel inventory and transaction have a one to many relationship, where a type of fuel can be **sold** in multiple transactions, and a transaction can only be associated with one type of fuel.
6. Employee and sale transaction have a one to many relationship, where an employee can provide multiple transactions to customers, and a transaction can only be associated with one employee.

## QUESTION 3

## ENTITY RELATIONSHIP DIAGRAM



## QUESTION 4

## PROJECT REPORT

## DATABASE SYSTEM FOR AIRPORT SHELL FILLING STATION


## ABSTRACT

In this project, a database to store the data about a filling station management is built using PostgreSQL . The database is built to store data about customers,employees, suppliers and fuel inventory and several other processes that is required for the efficient operation of the of the filling station. Some process in the database are also automated to enable efficient management and of the activities of the filling station so as to serve customers better.


## INTRODUCTION

Airport Shell is a large, filling station located near the Kotoka International airport. It's activities has gradually increase over time. Management has decided to automate all the activities of the filling station to enable the serve the customers better.

**Problem**: The is the need for digitization of processes and automation of processes so as to serve customers better .

**Solution**: Create a database to handle the manual processes and also automate majority of the activities of the filling station to so as to serve the customers better.

## DATABASE DESIGN AND METHODOLOGY

1. The problem which is the need to automate the activities of the Airport Shell filling station was understood.

2.A review was conducted of the current business processes, systems, and workflows used at the filling station. Necessary information was gathered pertaining to requirements and feedback from stakeholders, such as the station manager, employees, and customers.

3. Checked and analyzed the information to understand how well an automated database management system would improve the efficiency and accuracy of the filling station's operations.

4. A database management system was designed and built that includes all the necessary entities, attributes, and relationships from the entity relationship diagram.

5. The database was tested to ensure that it is scalable, robust, and can handle large volumes of data. This was done by running queries and inputting a considerable amount of data for testing purposes.

6.The database schema, including the ER diagram, table structures, and relationships, were all documented.

## DATABASE DESIGN AND IMPLEMENTATION

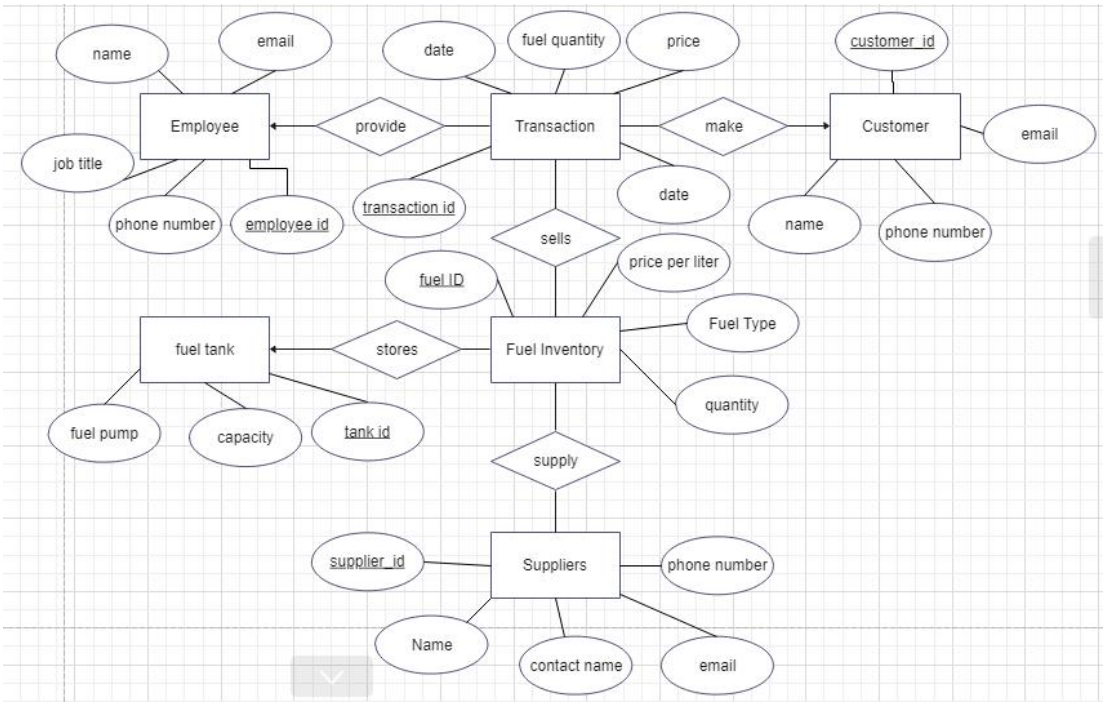
### Entities

- Fuel inventory
- Fuel tank
- Suppliers
- Employees
- Customers
- Transactions

### Relationship between Entities

| Entity | Relationship | Entity |
|---|---|---|
| Employee | Provides | Transaction |
| Customer | Makes | Transaction |
| Transaction | Sells | Fuel inventory |
| Fuel tank | Stores | Fuel inventory |
| Suppliers | Supply | Fuel inventory |

## ENTITY RELATIONSHIP DIAGRAM




## Database Schema

Fuel_tanks:( tank_id (primary key), fuel_type_id , capacity_in_liters)

Supplier:( supplier_id (primary key), supplier_name, contact_name, contact_email, contact_phone, tank_id )

Fuel_inventory: fuel_id (primary key), fuel_type, quantity_in_liters, price_per_liter, supplier_id )

Fuel_pumps:( pump_id (primary key), tank_id ))

Employee: (employee_id (primary key), first_name, last_name, job_title))

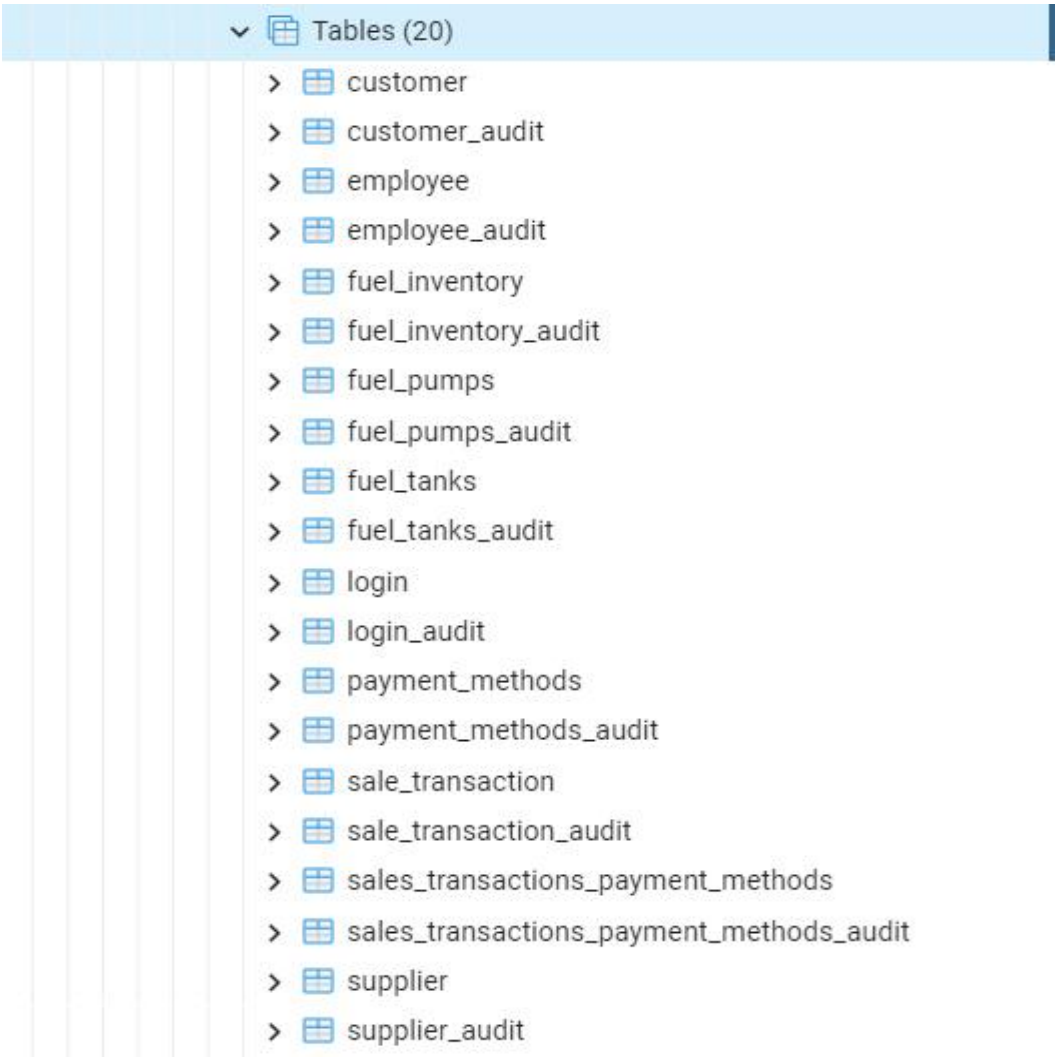
Customer:(customer_id (primary key), first_name, last_name, contact_number, email)

Sale_transaction: (transaction_id (primary key), transaction_date, fuel_id (foreign key to Fuel_inventory table), quantity_in_liters, price, customer_id (foreign key to Customer table))

## TABLES OR RELATIONS

All tables in database have an associated audit table to help track all changes the occur on a daily basis in the database. This help to ensure:

- Backup against losses of data
- Easy tracing of any illegal and unauthorized process that are performed in the database.




**Login table**

The login table will allow the database administrators to access the database only if they have the legitimate access.

Also in case an application will be created to allow customers to pay for fuel, check prices every day or any other services all they need to do is to create an account and

input their username and passwords to have access. So employees can also create an account and have access too.

```
CREATE TABLE login (
    id SERIAL PRIMARY KEY,
    username VARCHAR(50) UNIQUE NOT NULL,
    password VARCHAR(50) NOT NULL
);
```

```
SELECT *
FROM login
```

Data Output | Explain | Messages | Notifications

| | id [PK] integer | username character varying (50) | password character varying (50) |
|---|---|---|---|
| 1 | 1 | Kwabena.Asante | password123 |
| 2 | 2 | Adwoa.Agyei | qwerty456 |
| 3 | 3 | Kwame.Owusu | securepass |
| 4 | 4 | Esi.Amoah | pa$$word |
| 5 | 5 | Yaw.Nkrumah | p@ssw0rd! |
| 6 | 6 | Afia.Boateng | 12345678 |
| 7 | 7 | Kofi.Mensah | passw0rd |
| 8 | 8 | Ama.Addo | examplepass |
| 9 | 9 | Yaw.Akoto | password01 |

**FUEL INVENTORY**

Fuel inventory table keeps a record of the fuel products available in stock at a given point in time. It includes information such as the type of fuel, the quantity in liters,

and the price per liter. Maintaining a fuel inventory helps ensure that there is enough fuel to meet demand and that the fuel is being sold at the correct price. It also helps in tracking the amount of fuel used and the revenue generated by fuel sales.

```
1  SELECT *
2  FROM fuel_inventory
```

Data Output | Explain | Messages | Notifications

| | fuel_id [PK] character varying (20) | fuel_type character varying (50) | quantity_in_liters numeric (100) | price_per_liter numeric (10,2) | supplier_id integer |
|---|---|---|---|---|---|
| 1 | fuel01 | Gasoline | 1000 | 8.50 | 1 |
| 2 | fuel02 | Diesel | 750 | 10.80 | 1 |
| 3 | fuel03 | Propane | 500 | 7.20 | 2 |
| 4 | fuel04 | Ethanol | 250 | 6.80 | 3 |
| 5 | fuel05 | Methanol | 150 | 6.10 | 4 |
| 6 | fuel06 | Butanol | 190 | 9.60 | 5 |
| 7 | fuel07 | V power | 200 | 12.50 | 6 |
| 8 | fuel08 | Hydrogen | 300 | 3.50 | 7 |
| 9 | fuel09 | Natural Gas | 600 | 9.10 | 8 |
| 10 | fuel10 | Biofuel | 400 | 2.30 | 9 |

## FUEL TANKS

This table stores information about the various tanks for storing fuel and the current type of fuel in each tank.

```
1  SELECT *
2  FROM fuel_tanks
```

Data Output | Explain | Messages | Notifications

| | tank_id [PK] character varying (20) | fuel_type_id character varying (20) | capacity_in_liters numeric (100) |
|---|---|---|---|
| 1 | tank01 | fuel05 | 10000 |
| 2 | tank02 | fuel01 | 20000 |
| 3 | tank03 | fuel02 | 15000 |
| 4 | tank04 | fuel07 | 8000 |
| 5 | tank05 | fuel08 | 30000 |
| 6 | tank06 | fuel09 | 20000 |
| 7 | tank07 | fuel05 | 12000 |
| 8 | tank08 | fuel03 | 25000 |
| 9 | tank09 | fuel11 | 18000 |
| 10 | tank10 | fuel05 | 9000 |

## FUEL PUMPS

This table just indicates the fuel pump attached to each and every fuel tank. This will help to track any issues tha t arises due to bad or contaminated fuel.

```
SELECT *
FROM fuel_pumps
```

Data Output | Explain | Messages | Notifications

| | pump_id [PK] character varying (20) | tank_id character varying (20) |
|---|---|---|
| 1 | pump01 | tank01 |
| 2 | pump02 | tank02 |
| 3 | pump03 | tank03 |
| 4 | pump04 | tank04 |
| 5 | pump05 | tank05 |
| 6 | pump06 | tank06 |
| 7 | pump07 | tank07 |
| 8 | pump08 | tank08 |
| 9 | pump09 | tank09 |
| 10 | pump10 | tank10 |

## EMPLOYEE

Query Editor | Query History

```
SELECT *
FROM employee
```

Data Output | Explain | Messages | Notifications

| | employee_id [PK] integer | first_name character varying (50) | last_name character varying (50) | email character varying (100) | phone character varying (20) | job_title character varying (50) |
|---|---|---|---|---|---|---|
| 1 | 1 | Kwame | Agyapong | kwame.agyapong@gmail.com | 0241112223 | Fuel Attendant |
| 2 | 2 | Akosua | Mensah | akosua.mensah@yahoo.com | 0203334445 | Cashier |
| 3 | 3 | Yaw | Asante | yaw.asante@hotmail.com | 0265556667 | Assistant Manager |
| 4 | 4 | Ama | Boateng | ama.boateng@gmail.com | 0248889990 | Fuel Attendant |
| 5 | 5 | Kofi | Owusu | kofi.owusu@yahoo.com | 0207778881 | Cleaner |
| 6 | 6 | Abena | Tetteh | abena.tetteh@hotmail.com | 0243334442 | Cashier |
| 7 | 7 | Kwesi | Addo | kwesi.addo@gmail.com | 0264445556 | Fuel Attendant |
| 8 | 8 | Adwoa | Dankwa | adwoa.dankwa@yahoo.com | 0206667773 | Assistant Manager |
| 9 | 9 | Yaw | Mensah | yaw.mensah@hotmail.com | 0244445554 | Fuel Attendant |

This table keeps track of all employees and their roles in the company alongside some important details.

## CUSTOMER TABLE

This table helps to keep track of all customers who have ever made a transaction before, or those who have created an account on the company software.

```
1  SELECT *
2  FROM customer
```

Data Output | Explain | Messages | Notifications

| | customer_id [PK] integer | first_name character varying (50) | last_name character varying (50) | contact_number character varying (20) | email character varying (50) |
|---|---|---|---|---|---|
| 1 | 1 | Kwame | Agyemang | 0245879631 | kwame.agyemang@example.com |
| 2 | 2 | Ama | Boateng | 0556897412 | ama.boateng@example.com |
| 3 | 3 | Yaw | Danso | 0274968521 | yaw.danso@example.com |
| 4 | 4 | Akosua | Kwakye | 0541789624 | akosua.kwakye@example.com |
| 5 | 5 | Kwadwo | Kusi | 0204698532 | kwadwo.kusi@example.com |
| 6 | 6 | Adwoa | Mensah | 0265987412 | adwoa.mensah@example.com |
| 7 | 7 | Kofi | Owusu | 0556874596 | kofi.owusu@example.com |
| 8 | 8 | Ama | Tetteh | 0241589632 | ama.tetteh@example.com |
| 9 | 9 | Yaw | Yeboah | 0556798541 | yaw.yeboah@example.com |

## SUPPLIER

The supplier table keeps record of all companies that supply the filling station with fuel each and every time.

Query Editor | Query History

```
SELECT *
FROM supplier
```

Data Output | Explain | Messages | Notifications

| | supplier_id [PK] integer | supplier_name character varying (50) | contact_name character varying (50) | contact_email character varying (50) | contact_phone character varying (20) | tank_id character varying (20 |
|---|---|---|---|---|---|---|
| 1 | 1 | Ghana Oil Company | Kwame Adu | kwameadu@goc.com | +233-244-555-123 | tank01 |
| 2 | 2 | Ampofo Gas | Yaw Ampofo | yawampofo@ampofogas.com | +233-244-555-124 | tank02 |
| 3 | 3 | Oil City | Nana Yaa | nanayaa@oilcity.com | +233-244-555-125 | tank03 |
| 4 | 4 | Petronas Ghana | Kofi Asante | kofiasante@petronas.com | +233-244-555-126 | tank04 |
| 5 | 5 | Gazprom Ghana | Yuri Ivanov | yuriivanov@gazprom.com | +233-244-555-127 | tank05 |
| 6 | 6 | Black Star Energy | Ama Darko | amadarko@blackstarenergy.com | +233-244-555-128 | tank06 |
| 7 | 7 | Zen Petroleum | Kwesi Nyantakyi | kwesinyantakyi@zenpetroleum.com | +233-244-555-129 | tank07 |
| 8 | 8 | Star Oil | Emmanuel Amartey | emmanuelamartey@staroil.com | +233-244-555-130 | tank08 |
| 9 | 9 | Total Ghana | Grace Owusu | graceowusu@total.com | +233-244-555-131 | tank09 |

## TRANSACTION TABLE

This table keeps track of all transactions that have ever been made in the company by any customer. The price for each transaction is also calculated automatically.

```
678
679
680 SELECT * FROM sale_transaction
681
```

Data Output | Explain | Messages | Notifications

| | transaction_id [PK] integer | transaction_date timestamp without time zone | fuel_id character varying (20) | quantity_in_liters integer | price numeric (10,2) | customer_id integer |
|---|---|---|---|---|---|---|
| 1 | 1 | 2022-01-01 10:00:00 | fuel01 | 20 | 170.00 | 1 |
| 2 | 2 | 2022-01-02 11:00:00 | fuel02 | 15 | 162.00 | 2 |
| 3 | 3 | 2022-01-02 12:00:00 | fuel03 | 25 | 180.00 | 3 |
| 4 | 4 | 2022-01-03 13:00:00 | fuel04 | 5 | 34.00 | 4 |
| 5 | 5 | 2022-01-03 14:00:00 | fuel05 | 10 | 61.00 | 5 |
| 6 | 6 | 2022-01-04 15:00:00 | fuel06 | 30 | 288.00 | 6 |
| 7 | 7 | 2022-01-04 16:00:00 | fuel07 | 8 | 100.00 | 7 |
| 8 | 8 | 2022-01-05 17:00:00 | fuel08 | 20 | 70.00 | 8 |
| 9 | 9 | 2022-01-05 18:00:00 | fuel09 | 12 | 109.20 | 9 |
| 10 | 10 | 2022-01-06 19:00:00 | fuel10 | 15 | 34.50 | 10 |
| 11 | 11 | 2022-01-06 20:00:00 | fuel11 | 5 | 20.50 | 11 |

## PAYMENT METHOD TABLE AND SALES TRANSACTIONS PAYMENT METHOD

The payment method table keeps record of all modes of payment during a particular transaction, since not all payments are made physically there is the need to keep track of all payments.

Also, the sales transactions payments methods shows a list of all transactions and the method or mode of payment of that particular transaction.

Query Editor | Query History

```
1  SELECT *
2  FROM payment_methods
```

Data Output | Explain | Messages | Notifications

| | payment_method_id [PK] integer | payment_method_name character varying (50) |
|---|---|---|
| 1 | 1 | Credit Card |
| 2 | 2 | Debit Card |
| 3 | 3 | Cash |
| 4 | 4 | Mobile Money |
| 5 | 5 | Bank Transfer |
| | | |

```
SELECT *
FROM sales_transactions_payment_methods
```

Data Output | Explain | Messages | Notifications

| | sales_transaction_id integer | payment_method_id integer |
|---|---|---|
| 1 | 1 | 1 |
| 2 | 2 | 2 |
| 3 | 3 | 1 |
| 4 | 4 | 3 |
| 5 | 5 | 4 |
| 6 | 6 | 2 |
| 7 | 7 | 3 |
| 8 | 8 | 1 |
| 9 | 9 | 4 |
| 10 | 10 | 2 |

## AUDIT TABLES OR RELATIONS

Due to the loss and tampering of data in the database an audit table was created for each and every relation in the database, this will help serve as a backup.

Trigger functions and triggers are placed on each table and connected to the audit tables such that whenever a deletion or update is performed on the database, the previous and current information will be automatically sent to the audit tables.

## AUTOMATIC FUNCTIONS AND PROCESSES

### Auto update fuel quantity

This function automatically updates the fuel inventory table whenever a new transaction is made in the sales transaction table.

```
CREATE OR REPLACE FUNCTION update_fuel_inventory()

RETURNS TRIGGER AS $$

BEGIN

    UPDATE fuel_inventory

    SET quantity = quantity – NEW.quantity_in_liters

    WHERE fuel_id = NEW.fuel_id;
```

```
    RETURN NEW;
END;
$$ LANGUAGE plpgsql;


CREATE TRIGGER update_fuel_inventory_trigger
AFTER INSERT ON sale_transaction
FOR EACH ROW
EXECUTE PROCEDURE update_fuel_inventory();
```

**Automated Fuel Running low Alert**

This function will show an alert message when ever a particular fuel is running low.

```
CREATE OR REPLACE FUNCTION show_inventory_alert()
RETURNS TRIGGER AS $$
DECLARE
    inventory_level NUMERIC;
    fuel_type TEXT;
BEGIN
    SELECT quantity_in_liters, fuel_Type INTO inventory_level, fuel_type FROM fuel_inventory WHERE fuel_id = NEW.fuel_id;
    IF inventory_level < 100 THEN
        RAISE NOTICE 'Fuel inventory is running low for %', fuel_type;
    END IF;
    RETURN NEW;
END;
```

$$ LANGUAGE plpgsql;

CREATE TRIGGER inventory_alert_trigger

AFTER UPDATE ON fuel_inventory

FOR EACH ROW

WHEN (NEW.quantity_in_liters < 100)

EXECUTE PROCEDURE show_inventory_alert();

```
--Automatic processes testing

/*To test the modified `show_inventory_alert()` function, an update was made in
`quantity_in_liters` for a fuel type and  to check if the alert message includes the correct fuel
UPDATE fuel_inventory
SET quantity_in_liters = 50
WHERE fuel_type = 'Diesel';

```

Data Output | Explain | Messages | Notifications

```
NOTICE:  Fuel inventory is running low for Diesel
UPDATE 1

Query returned successfully in 103 msec.
```

With such a function in place a build up can be made in association with another programming language such as python to automatically send an email to a supplier whenever a particular fuel is running low.

Another function can be created with a similar format to automatically send the price per liter of each fuel type to customers whenever a price is changed.

**Automated calculation of fuel price for number of liters in sales transaction table**

CREATE TRIGGER sale_transactions_audit_trigger

AFTER INSERT OR UPDATE OR DELETE ON sale_transaction

FOR EACH ROW

EXECUTE PROCEDURE sale_transactions_audit();

```
CREATE OR REPLACE FUNCTION update_sale_transaction_price()
RETURNS TRIGGER AS $$
BEGIN
   UPDATE sale_transaction
   SET price = NEW.quantity_in_liters * (
            SELECT price_per_liter
            FROM fuel_inventory
            WHERE fuel_id = NEW.fuel_id
        )
   WHERE transaction_id = NEW.transaction_id;

   RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER update_sale_transaction_price_trigger
AFTER INSERT ON sale_transaction
FOR EACH ROW
EXECUTE FUNCTION update_sale_transaction_price();
```

**Quick Insert Functions**

**This function is supposed to allow the database admin to easily add an new customer to the database:**

```
CREATE OR REPLACE FUNCTION add_customer(
  First_name VARCHAR(50),
  Last_name VARCHAR(50),
  Contact_number VARCHAR(20),
  email VARCHAR(50)
)
RETURNS VOID AS $$
BEGIN
  INSERT INTO customer(first_name, last_name, contact_number, email)
  VALUES (first_name, last_name, contact_number, email);
END;
$$ LANGUAGE plpgsql;
```

**This function is supposed to allow the database admin to easily add an new employee to the database:**

```
CREATE OR REPLACE FUNCTION add_employee(
  first_name VARCHAR(50),
  last_name VARCHAR(50),
  email VARCHAR(100),
  phone VARCHAR(20),
  job_title VARCHAR(50)
)
RETURNS VOID AS $$
BEGIN
```

INSERT INTO Employees(first_name, last_name, email, phone, job_title)

VALUES (first_name, last_name, email, phone, job_title);

END;

$$ LANGUAGE plpgsql;

TESTS OF ALL THESE FUNCTIONS ARE INCLUDED IN THE QUERY SCRIPT FOR VISUAL ILLUSTRATION.

## TESTING OF DATABASE BY RUNNING QUERIES

**--List all the fuel inventory items along with their quantity and supplier**

```
71  --List all the fuel inventory items along with their quantity and supplier
72  SELECT fuel_id, fuel_type, quantity_in_liters, supplier_name
73  FROM fuel_inventory
74  INNER JOIN supplier ON fuel_inventory.supplier_id = supplier.supplier_id;
75
76
```

Data Output | Explain | Messages | Notifications

| | fuel_id character varying (20) | fuel_type character varying (50) | quantity_in_liters numeric (100) | supplier_name character varying (50) |
|---|---|---|---|---|
| 1 | fuel15 | LPG | 700 | Springfield Energy |
| 2 | fuel24 | Diesel | 150 | Ghana Oil Company |
| 3 | fuel25 | Gasoline | 980 | Ghana Oil Company |
| 4 | fuel26 | Propane | 475 | Ampofo Gas |
| 5 | fuel27 | Ethanol | 245 | Oil City |
| 6 | fuel28 | Methanol | 140 | Petronas Ghana |
| 7 | fuel29 | Butanol | 160 | Gazprom Ghana |
| 8 | fuel30 | V power | 192 | Black Star Energy |
| 9 | fuel31 | Hydrogen | 280 | Zen Petroleum |
| 10 | fuel32 | Natural Gas | 588 | Star Oil |
| 11 | fuel33 | Biofuel | 385 | Total Ghana |

SELECT fuel_id, fuel_type, quantity_in_liters, supplier_name

FROM fuel_inventory

INNER JOIN supplier ON fuel_inventory.supplier_id = supplier.supplier_id;

**--List all the sale transactions along with the customer and payment method used**

```
76
77 --List all the sale transactions along with the customer and payment method used
78 SELECT transaction_id, transaction_date, fuel_id, quantity_in_liters, price, first_name, last_name, pa
79 FROM sale_transaction
80 INNER JOIN customer ON sale_transaction.customer_id = customer.customer_id
81 INNER JOIN sales_transactions_payment_methods ON sale_transaction.transaction_id = sales_transactions_
82 INNER JOIN payment_methods ON sales_transactions_payment_methods.payment_method_id = payment_methods.p
83
84
```

Data Output | Explain | Messages | Notifications

| | transaction_id integer | transaction_date timestamp without time zone | fuel_id character varying (20) | quantity_in_liters integer | price numeric (10,2) | first_name character varying (50) | last_name character varying (50) |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 2022-01-01 10:00:00 | fuel01 | 20 | 170.00 | Kwame | Agyemang |
| 2 | 2 | 2022-01-02 11:00:00 | fuel02 | 15 | 162.00 | Ama | Boateng |
| 3 | 3 | 2022-01-02 12:00:00 | fuel03 | 25 | 180.00 | Yaw | Danso |
| 4 | 4 | 2022-01-03 13:00:00 | fuel04 | 5 | 34.00 | Akosua | Kwakye |
| 5 | 5 | 2022-01-03 14:00:00 | fuel05 | 10 | 61.00 | Kwadwo | Kusi |
| 6 | 6 | 2022-01-04 15:00:00 | fuel06 | 30 | 288.00 | Adwoa | Mensah |
| 7 | 7 | 2022-01-04 16:00:00 | fuel07 | 8 | 100.00 | Kofi | Owusu |
| 8 | 8 | 2022-01-05 17:00:00 | fuel08 | 20 | 70.00 | Ama | Tetteh |
| 9 | 9 | 2022-01-05 18:00:00 | fuel09 | 12 | 109.20 | Yaw | Yeboah |

## --Find all fuel inventory items with their associated supplier and tank capacity:

```
107 --Find all fuel inventory items with their associated supplier and tank capacity:
108 SELECT f.fuel_id, f.fuel_type, f.quantity_in_liters, f.price_per_liter, s.supplier_name, t.capacity_in
109 FROM fuel_inventory as f
110 JOIN supplier as s ON f.supplier_id = s.supplier_id
111 JOIN Fuel_tanks t ON s.tank_id = t.tank_id;
112
113
114 --Find all fuel pump IDs and their associated tank IDs:
115
```

Data Output | Explain | Messages | Notifications

| | fuel_id character varying (20) | fuel_type character varying (50) | quantity_in_liters numeric (100) | price_per_liter numeric (10,2) | supplier_name character varying (50) | capacity_in_liters numeric (100) |
|---|---|---|---|---|---|---|
| 1 | fuel15 | LPG | 700 | 6.80 | Springfield Energy | 28000 |
| 2 | fuel24 | Diesel | 150 | 10.80 | Ghana Oil Company | 10000 |
| 3 | fuel25 | Gasoline | 980 | 8.50 | Ghana Oil Company | 10000 |
| 4 | fuel26 | Propane | 475 | 7.20 | Ampofo Gas | 20000 |
| 5 | fuel27 | Ethanol | 245 | 6.80 | Oil City | 15000 |
| 6 | fuel28 | Methanol | 140 | 6.10 | Petronas Ghana | 8000 |
| 7 | fuel29 | Butanol | 160 | 9.60 | Gazprom Ghana | 30000 |
| 8 | fuel30 | V power | 192 | 12.50 | Black Star Energy | 20000 |
| 9 | fuel31 | Hydrogen | 280 | 3.50 | Zen Petroleum | 12000 |

## Find all sale transactions with their associated customer names and payment methods

Query Editor | Query History

```
123
124 --Find all sale transactions with their associated customer names and payment methods:
125 SELECT s.transaction_id, s.transaction_date, c.first_name, c.last_name, p.payment_method_name
126 FROM sale_transaction as s
127 JOIN customer as c ON s.customer_id = c.customer_id
128 JOIN Sales_transactions_Payment_methods stpm ON s.transaction_id = stpm.sales_transaction_id
129 JOIN Payment_methods p ON stpm.payment_method_id = p.payment_method_id;
130
131
```

Data Output | Explain | Messages | Notifications

| | transaction_id integer | transaction_date timestamp without time zone | first_name character varying (50) | last_name character varying (50) | payment_method_name character varying (50) |
|---|---|---|---|---|---|
| 1 | 1 | 2022-01-01 10:00:00 | Kwame | Agyemang | Credit Card |
| 2 | 2 | 2022-01-02 11:00:00 | Ama | Boateng | Debit Card |
| 3 | 3 | 2022-01-02 12:00:00 | Yaw | Danso | Credit Card |
| 4 | 4 | 2022-01-03 13:00:00 | Akosua | Kwakye | Cash |
| 5 | 5 | 2022-01-03 14:00:00 | Kwadwo | Kusi | Mobile Money |
| 6 | 6 | 2022-01-04 15:00:00 | Adwoa | Mensah | Debit Card |
| 7 | 7 | 2022-01-04 16:00:00 | Kofi | Owusu | Cash |
| 8 | 8 | 2022-01-05 17:00:00 | Ama | Tetteh | Credit Card |
| 9 | 9 | 2022-01-05 18:00:00 | Yaw | Yeboah | Mobile Money |

**More query testing is done in the query script file.**

## CONCLUSION

In conclusion, the current database will be able to solve almost all the needs of Airport Shell filling station. All information will be tracked, and there is a backup for any loss of data or issues that may arise at any point in time. Above there are several automatic functions and processes in the database that will help serve customers efficiently and better.