**Table of Content**

**Task 1 – Description of Business1**

1.1 Business Overview 1

1.2 Business Operations and Data Requirement 2

1.3 Project and Database Scope2

**Task 2 – ER and Data Dictionary3**

2.1 Entity Relationship4

2.2 Data Dictionary4

2.3 Normalization10

2.4 Justification of Normalization11

2.5 Integrity Constraints12

**Task 3 – Script to Create Table Structure14**

3.1 SQL Script used to Create Database14

3.2 Description of Database Script23

3.3 Errors Encountered23

**Task 4 – Data Population**2**4**

4.1 Data Population Screenshots24

4.2 Description of Data Population 30

4.3 Errors Encountered30

**Task 5 - Enhancement33**

5.1 Enhancement33

**Task 6 – SQL Reports34**

6.1 Useful SQL Reports34

**Task 7 -Distributed Database Option38**

7.1 Why Distributed Database38

**Task 8 – Reflective Practice39**

8.1 Reflective Practice using Rolfe, G (2001) Model39

**1.0 Task 1 – Description of Business**

Give a written overview of your organisation.

You should include an outline of the context in which they operate. For example, a college would operate in an environment in competition with other colleges.

You should outline their day-to-day operations, their data requirements and the types of transactions they carry out. You might include scans and/or diagrams of any documents you have gathered, which should be suitably anonymised so as not to show any personal data that might be included. You should explain the purpose of these documents in the context of the organisation.

You should outline the scope of the database project that you will undertake. This should be suitably ambitious enough to include at least ten entities in the data model. You should make clear what will be included within the scope of the project and what will not be included. This discussion of scope should include both data and operational issues.

**1.1 Business Overview**

Metalgate Solutions is an IT company situated at Anthony village, Lagos, Nigeria. They deal with retailing of various kinds of computer and computer accessories such as camera, mouse, keyboard etc. They also provide services such as repair of computer systems, installation of software and CCTV.

Through tough competition the company must run efficiently and reduce cost. This includes using updated information in its purchasing decisions, inventory control, store stocking, buying trending products and most importantly cater to the daily needs of customers and provide them with a well-priced, quality and reliable products and services everyday

**1.2 Business Operations and Data Requirement**

It requires various data to carry out the daily operations which involves catering to customers need, generating invoice based on customer’s request, generating quotation for services, the cashier processing customers invoice and issuing receipt to customer after a successful transaction, inventory control and repair/installation of systems. The required data includes item information, employee information, inventory control, customer information, checkout activities and purchase activities.

**1.3 Project & Database Scope**

This project will focus on creating a database that will effectively keep records of staffs, customers and the day to day activities of Metalgate Solutions.

A CUSTOMER can order for more than one TEMS on a single INVOICE. A QUOTATION is generated for each SERVICE rendered to customer. SERVICE has a SERVICE\_TYPE that indicate the type of service

Each EMPLOYEE has a DEPATMENT and STATUS, and performs operations like generate INVOICE, QUOTATION and RECEIPT for CUSTOMERS, adding TEMS, and making a PURCHASE

Every TEMS has a CATEGORY, inventory (ITEM\_UPDATE) is updated when a PURCHASE and order is made. items STATUS shows weather the items available, low in stock or not available

Every RECEIPT has a RECEIPT\_ TYPE indicates a successful transaction of a purchase or service.

Every PURCHASE is linked to one or more SUPPLIER.

**2.0 TASK 2 - ER and Data Dictionary**

Produce an entity relationship diagram for your organisation and an accompanying data dictionary. Your entity relationship diagram should include at least TEN (10) entities. You should use the UML format.

Briefly explain the purpose of normalisation, and clearly justify why each of your entities is in 3NF.

Using an example from your design, explain how normalisation solves the problem of update anomalies

Define the integrity constraints on the tables in your design. You should clearly include Domain constraints, Business rules (Table constraints) and Propagation constraints in your data dictionary.

**2.1 Entity Relationship**

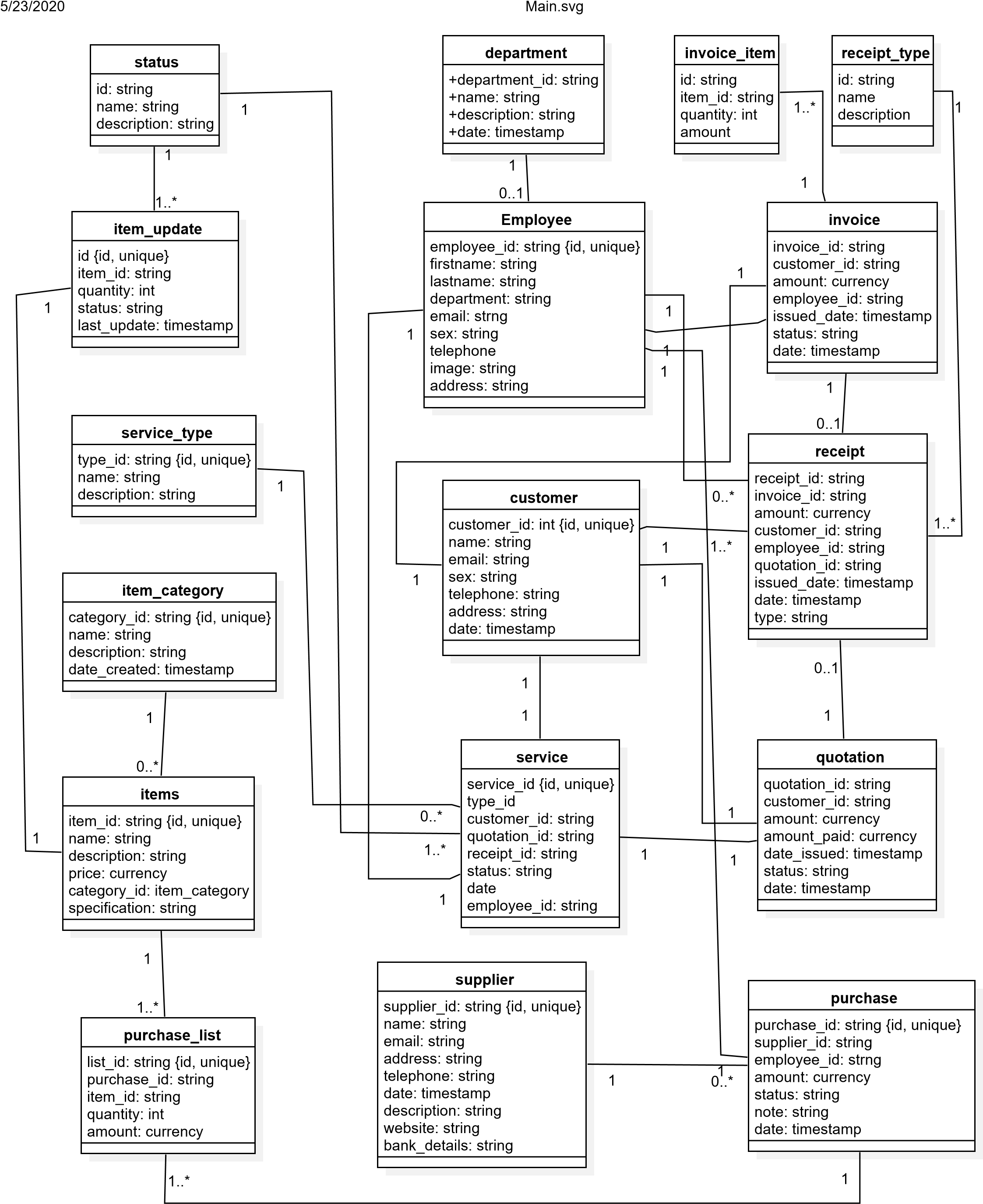


Fig 2.0 entity relationship using uml class diagram

**2.2 Data Dictionary**

**Customer**

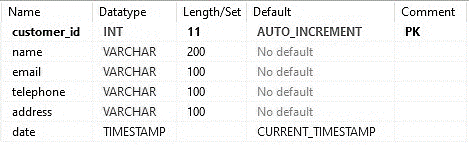
****

Fig 2.1 data dictionary of customers entity

**Employee**

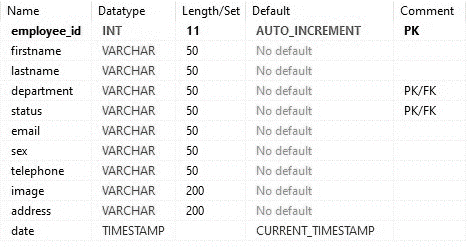
****

Fig 2.2 data dictionary of employee entity

**department**

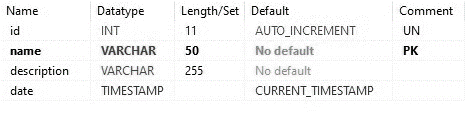
****

Fig 2.3 data dictionary of department entity

**Status**

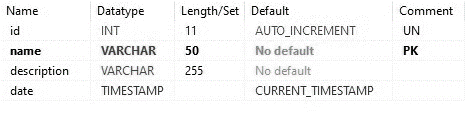
****

Fig 2.4 data dictionary of status entity, with name as the primary key

**Supplier**

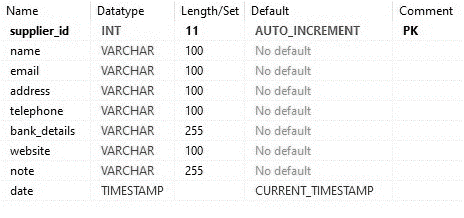
****

Fig 2.5 data dictionary of supplier entity

**Service\_type**

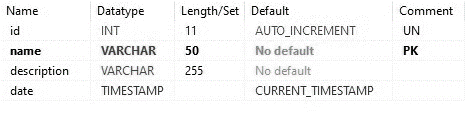
****

Fig 2.6 data dictionary of service\_type entity

**Item\_category**

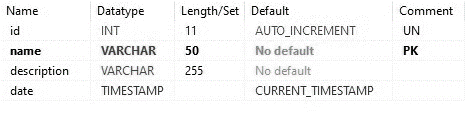
****

Fig 2.7 data dictionary of Item\_category entity

**Quotation**

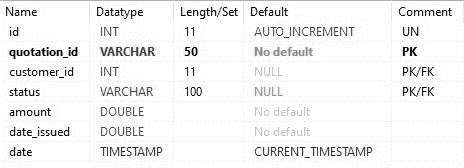


Fig 2.8 data dictionary of Quotation entity

**Receipt\_type**

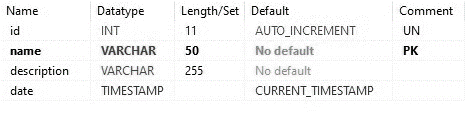
****

Fig 2.9 data dictionary Receipt\_type entity

**Receipt**

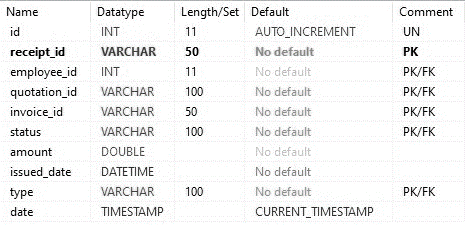


Fig 2.10 data dictionary of Receipt entity

**Service**

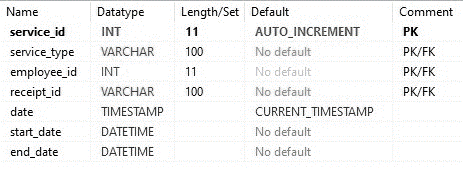
****

Fig 2.11 data dictionary of Service entity

**Items**

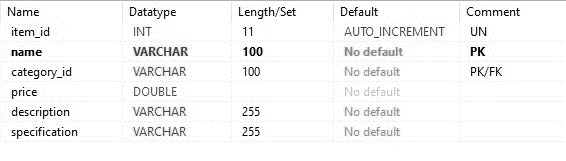


Fig 2.12 data dictionary of Items entity

**Item\_update**

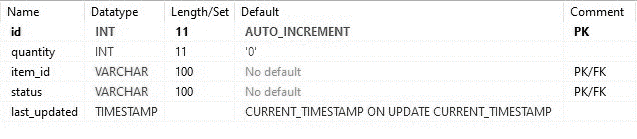


Fig 2.13 data dictionary of Receipt entity

**invoice**

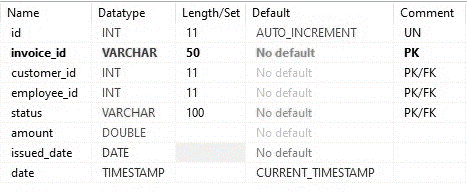


Fig 2.14 data dictionary of Invoice entity

**Invoice\_item**

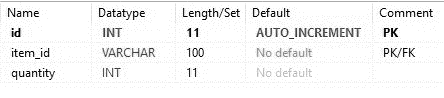


Fig 2.15 data dictionary of Invoice\_item entity

**purchase**

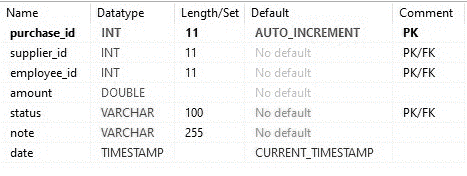
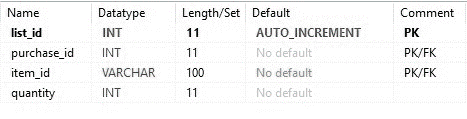


Fig 2.16 data dictionary of Purchase entity

**Purchase\_list**

Fig 2.17 data dictionary of Purchase\_list entity

**2.3 Normalisation**

Normalisation is majorly adopted in this design to standardize the database:prevent insertion, update, and deletion anomalies that could lead to a loss of data integrity.

The forms of normalisation used in this application are First normal form, second normal form and third normal form. A relation is in 3NF if it conforms to 2NF, and also every column that is not a primary key is only dependent on the whole of the primary key. Fig 2.19 below helps justify the use of 3NF relation in this design

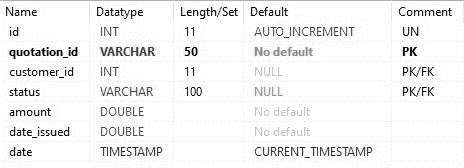


Fig 2.18 Justification of 3NF relation using Quotation entity

Fig 2.19 3NF relation Heireichy that exists between quotation, customer and status entity

Looking at the quotation table. The customer\_id column is linked to the customers table, and also the status of the quotation (status column) is derived from the status table. This relation makes the database simplified, flexible and eradicates update anomalies.

**2.4 Justification of Normalization**

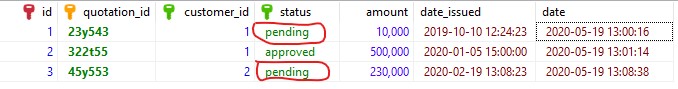
Using the quotation and invoice table as a case study where we have status column as a foreign key whose parent entity is the status table.  
  


Fig 2.20 Quotation entity to illustrates normalisation

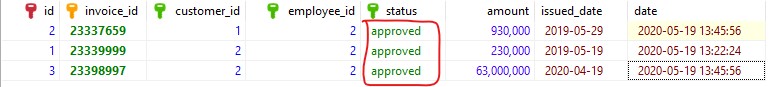


Fig 2.21 Invoice table to illustrates normalisation

Fig 2.20 and Fig 2.21 shows the quotation table and invoice table respectively. In a situation where the company wants to change the word “pending” as a status to “processing”. without proper normalisation, the process would be very tedious and prone to error.

Because we adopted normalisation, all we have to do is change the name in the parent table (status entity) below and it will reflect in all the table that referenced it.



Fig 1.21 Status table to illustrates normalisation

**2.5 Integrity Constraints**

**2.5.1 Entity Integrity Constraint**

States that no primary key value can be NULL. This is because the primary key value is used to identify individual tuples in a relation. Having NULL values for the primary key implies that we cannot identify some tuples.

**2.5.2 Primary Key and Unique Key Constraints**

An entity type usually has an attribute whose values are distinct for each individual entity in the entity set. Such an attribute is called a key attribute, and its values can be used to identify each entity uniquely.

**2.5.3 Propagation Constraint**

It details what should happen to a related table when a user update or delete a row or rows of a target table. It is used to maintain the consistency among tuples in the two relations.

**2.5.4 Check Constraints**

It is a condition that defines valid data when adding or updating an entry in a table of a relational database

**3.0 Task 3 – Scripts to create table structures (15 marks)**

Show the SQL scripts that you have used to create your database in a suitable database environment. You should include scripts for tables, columns, primary and foreign keys and any other database objects that you use. You should include screen shots of the scripts running within the database environment. For high marks, your scripts should implement integrity constraints you identified in Task 2.

You should provide an explanation of how you developed the scripts, order of running,

and any issues encountered with them.

**3.1 SQL Scripts Used to Create Database**

**Creating database using the company’s name**

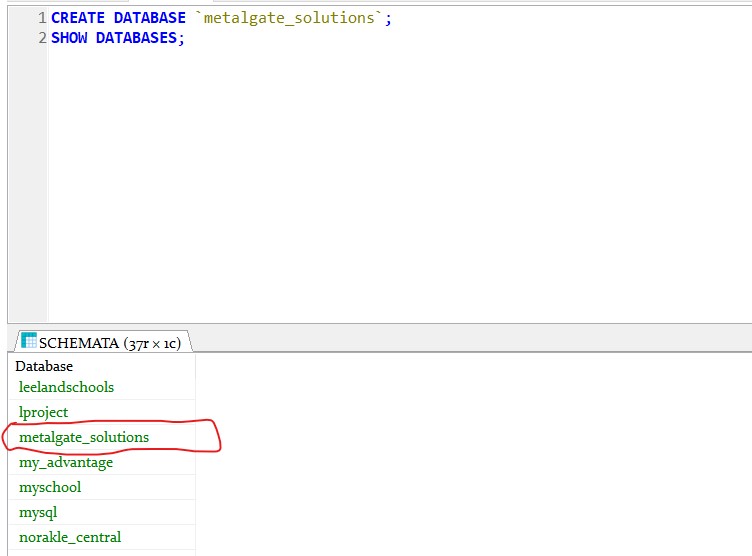
****

Fig 3.0 database creation SQL statement screenshot

**Customers table SQL script**

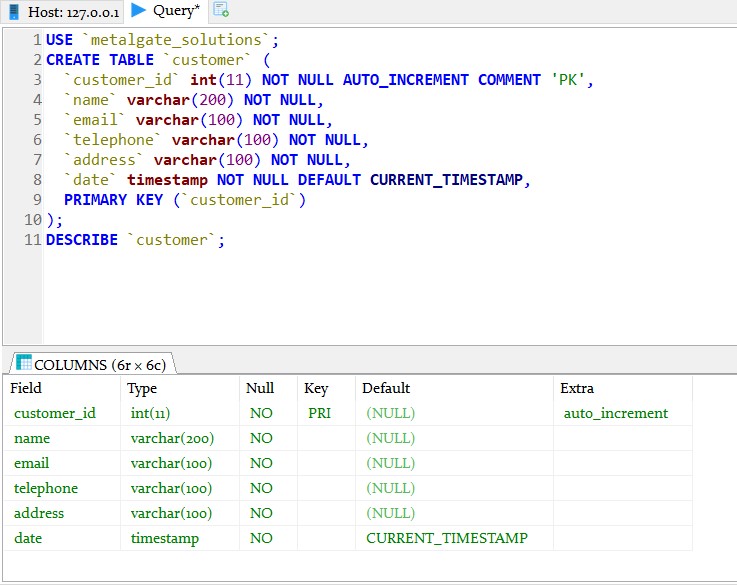
****

Fig 3.1 Screenshot of SQL statement for customers table creation

**Department table SQL script**

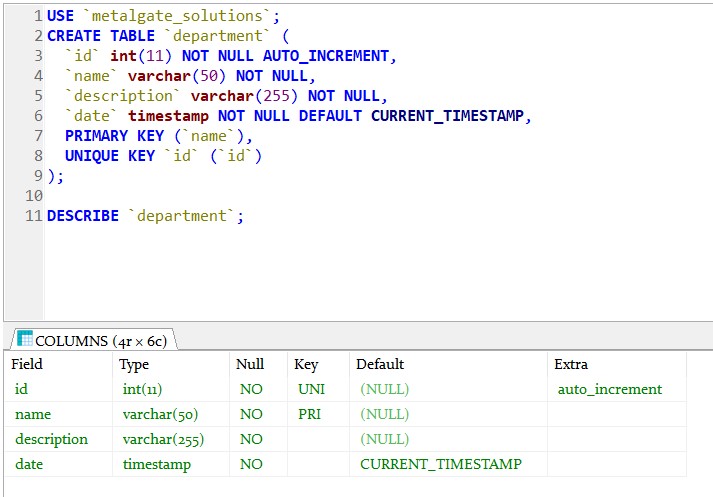
****

Fig 3.2 Screenshot of SQL statement for department table creation

**Status table**

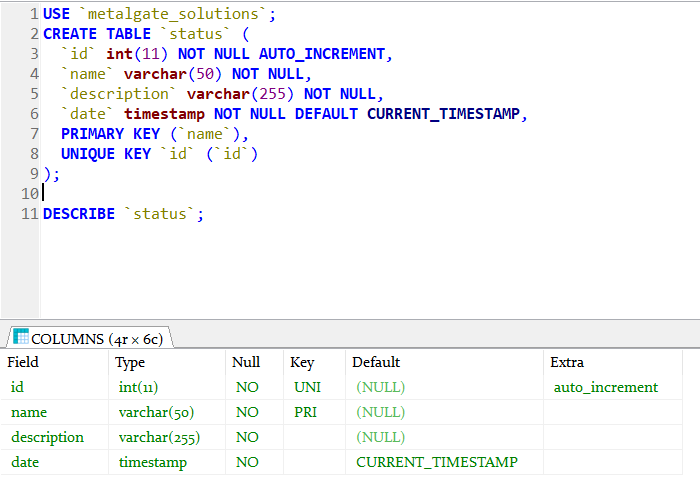
****

Fig 3.3 Screenshot of SQL statement for status table creation

**Employee table SQL script**

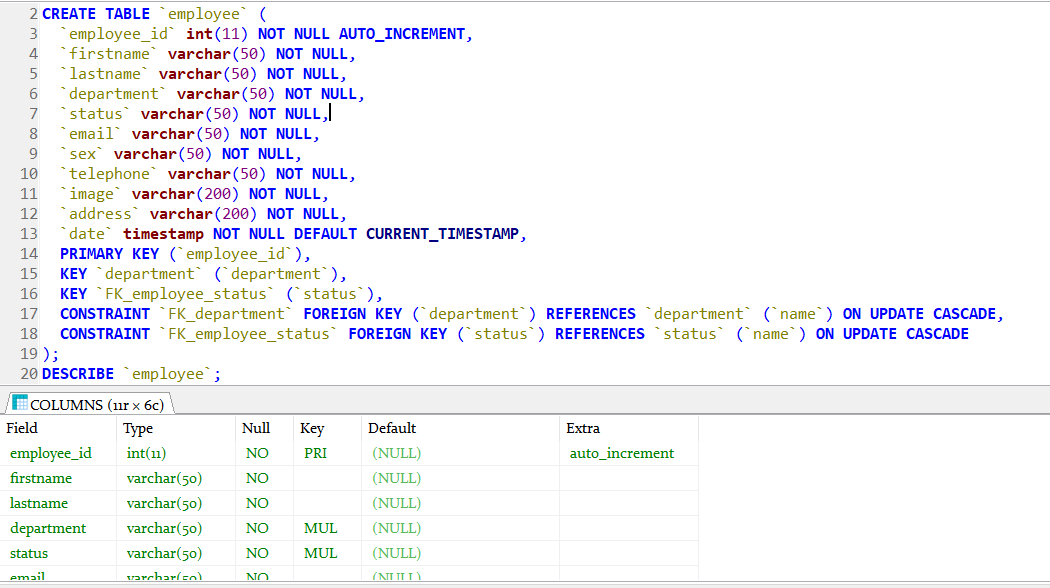


Fig 3.4 Screenshot of SQL statement for creating employee table

**Item\_category**

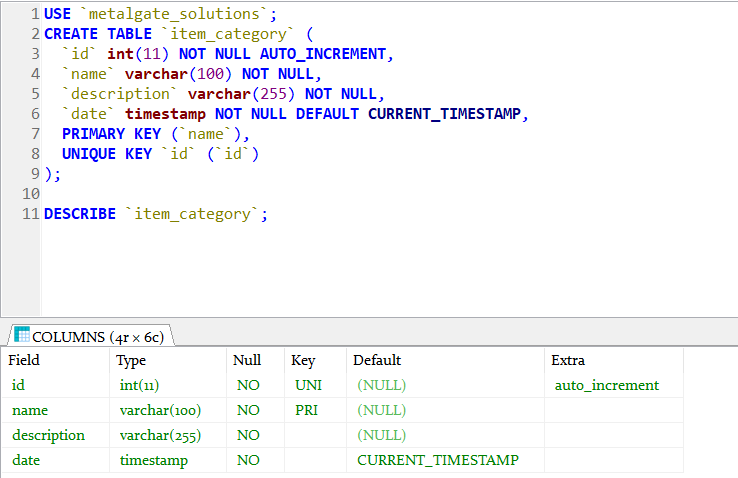


Fig 3.5 Screenshot of SQL statement for creating item\_category table

**Items table**

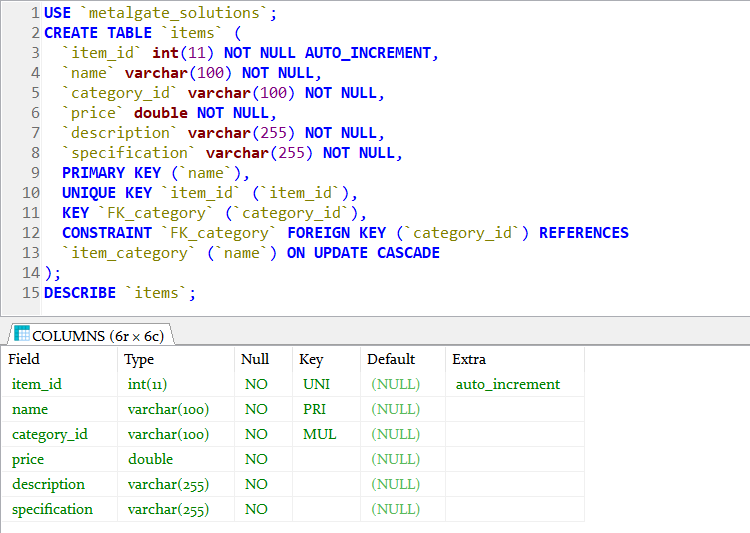


Fig 3.6 Screenshot of SQL statement for creating items table

**Item\_update**

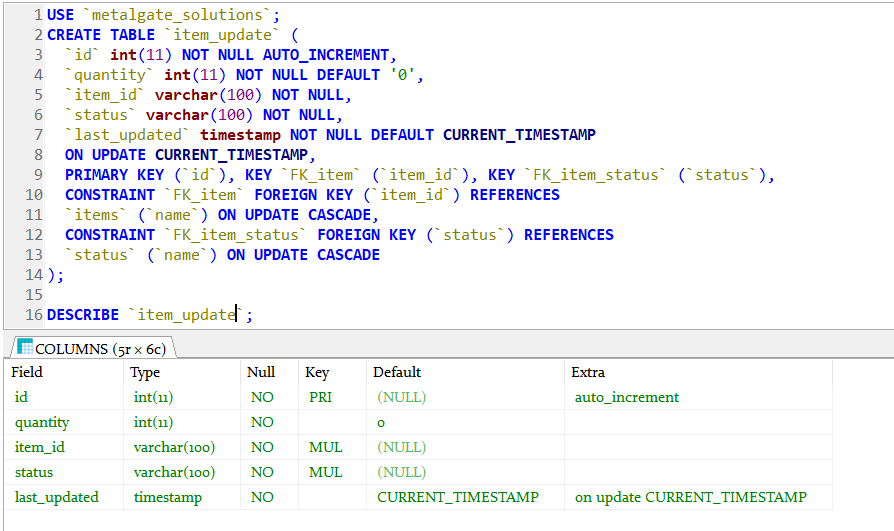


Fig 3.7 Screenshot of SQL statement for creating item\_update table

**Invoice table SQL script**

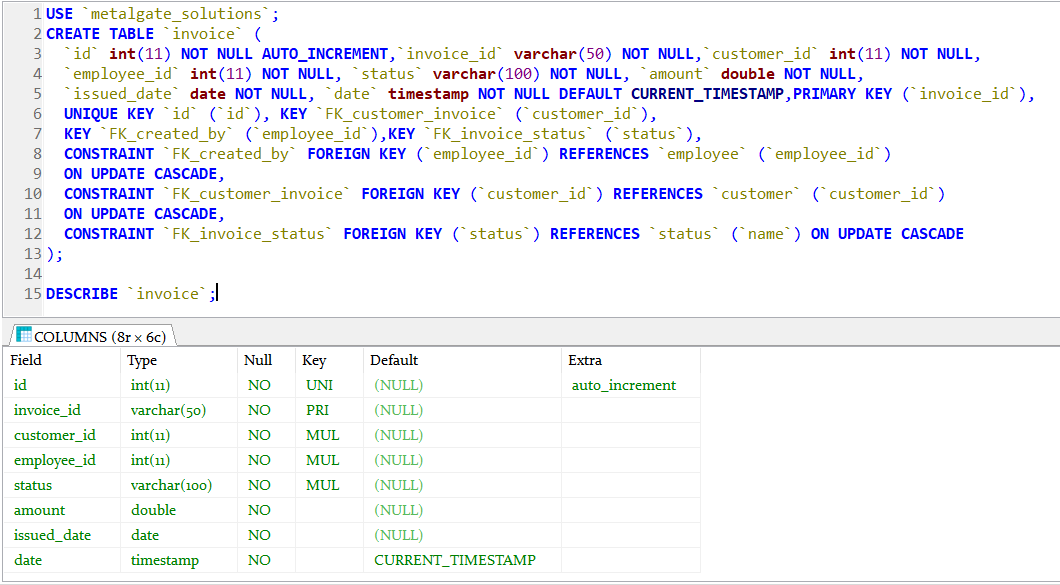


Fig 3.8 Screenshot of SQL statement for creating invoice table

**Invoice\_item table**

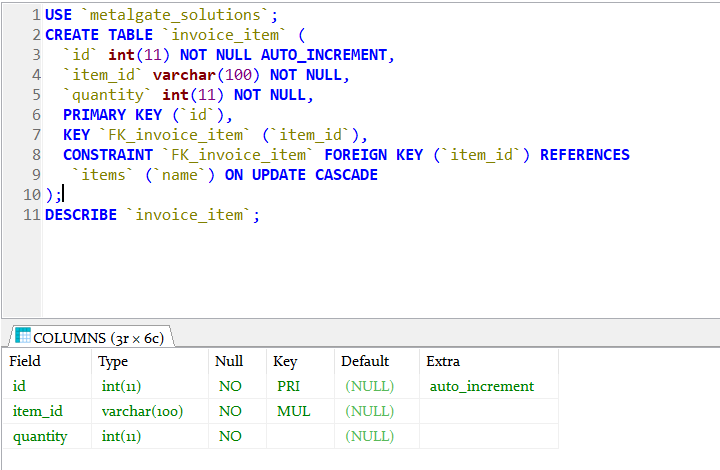
****

Fig 3.9 Screenshot of SQL statement for invoice\_item employee table

**Supplier**

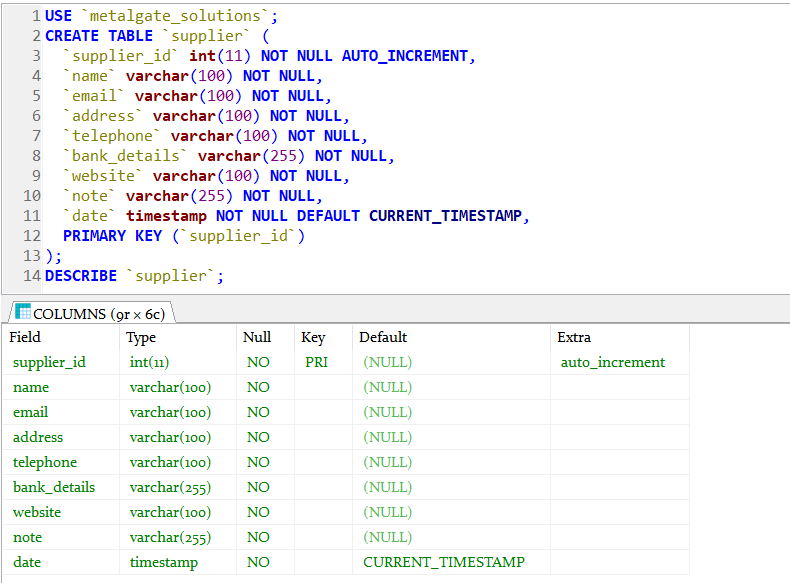
****

Fig 3.10 Screenshot of SQL statement for creating supplier table

**Purchase**

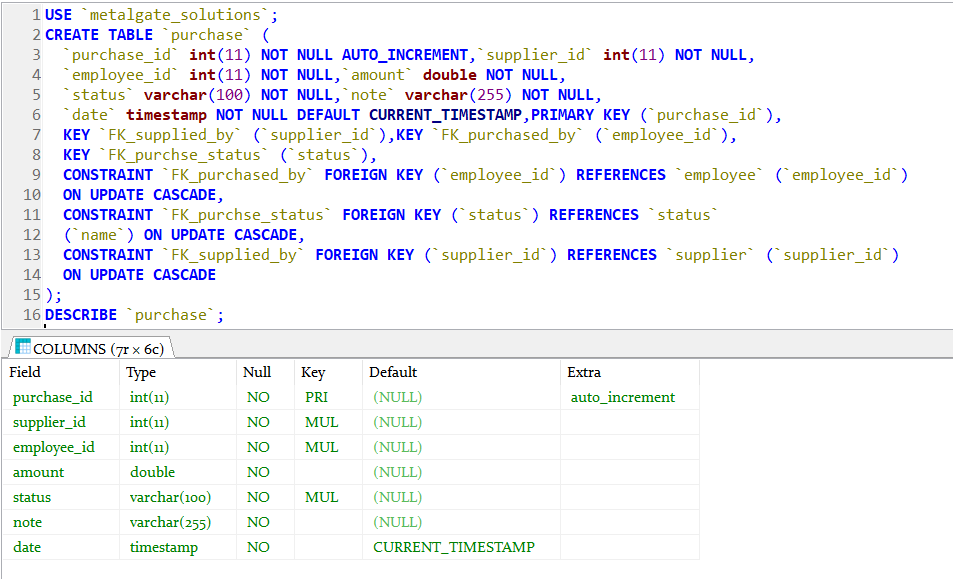
****

Fig 3.11 Screenshot of SQL statement for creating purchase table

**Purchase\_list**

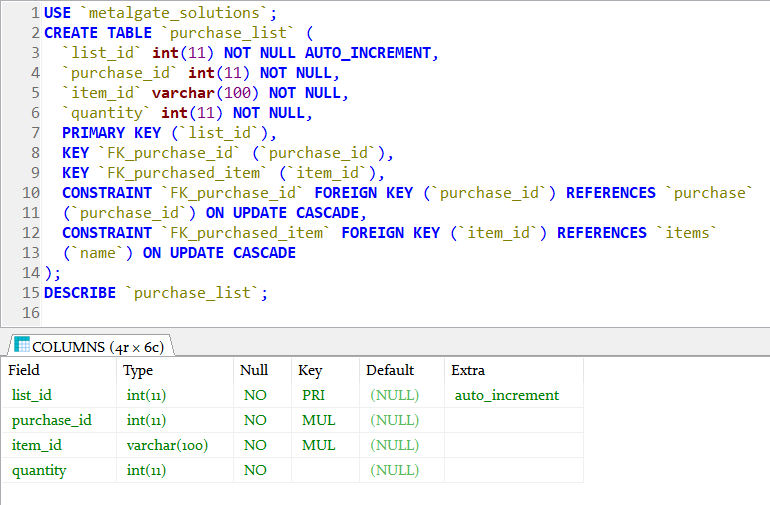
****

Fig 3.12 Screenshot of SQL statement for creating purchase\_list table

**Quotation**

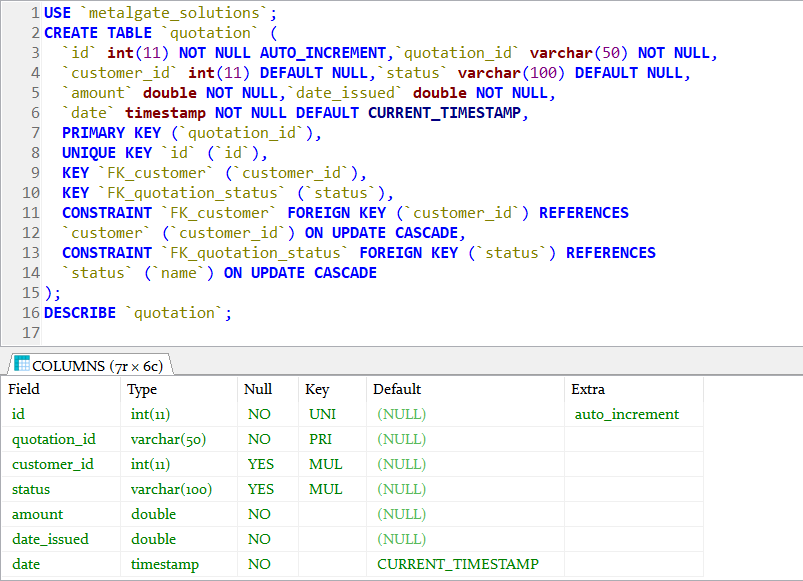
****

Fig 3.13 Screenshot of SQL statement for creating quotation table

**Receipt\_type**

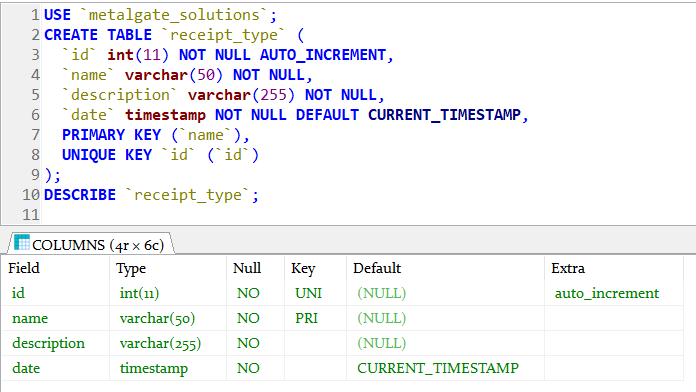
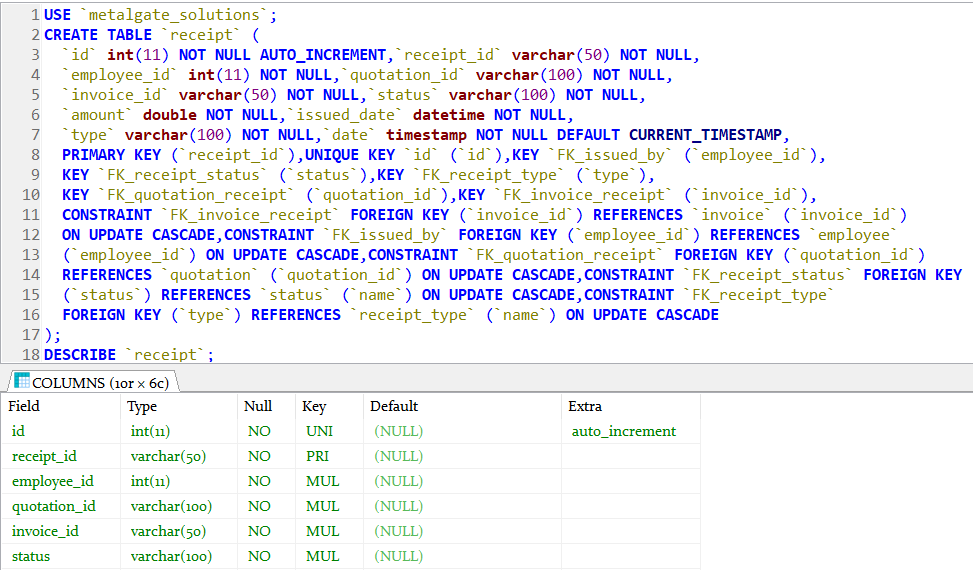
****

Fig 3.14 Screenshot of SQL statement for creating receipt\_type table

**Receipt**

****

****

Fig 3.15 Screenshot of SQL statement for creating receipt table

**Service\_type**

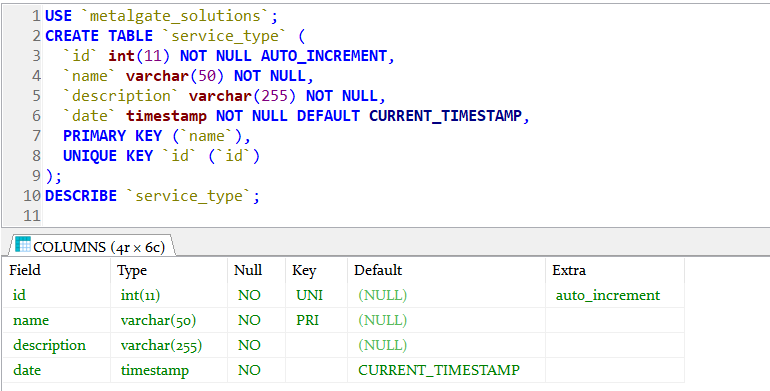
****

Fig 3.16 Screenshot of SQL statement for creating service\_type table

**Service**

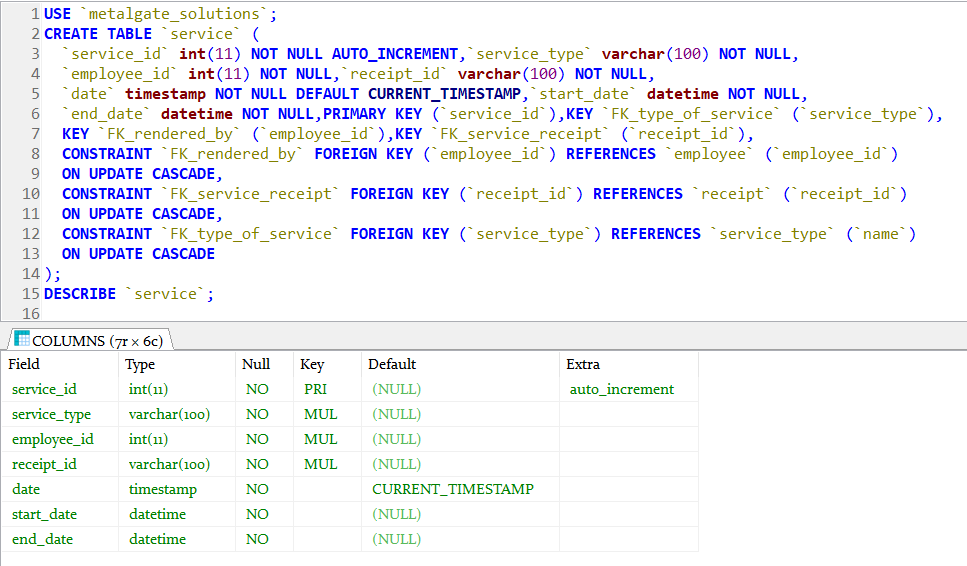
****

Fig 3.17 Screenshot of SQL statement for creating service table

**3.2 Description of Database Creation Script**

First, I created the 1NF tables which are CUSTOMERS, DEPARTMENT, STATUS, SERVICE\_TYPE, RECEIPT\_TYPE, ITEM\_CATEGORY AND SUPPLIER table because they are not dependent on any table. Follow by the EMPLOYEE table, which is dependent of the DEPARTMENT and STATUS table. Then the ITEMS, PURCHASE and PURCHASE\_LIST table respectively, because the PURCHASE\_LIST is dependent on ITEMS and PURCHASE table. Then the QUOTATION, INVOICE, INVOICE\_ITEM and ITEM\_UPDATE table, and lastly, I created the RECEIPT and SERVICE table, respectively because the service table is dependent on the RECEIPT.

**3.3 Error encountered**

The major problems encountered during the database creation is missing semi-column. Another problem encountered was omission of primary key, and lastly creating duplicate foreign key name

**4.0 Task 4 – Data population (10 marks)**

You should create suitable data for your database. Show the SQL insert scripts that you have used to populate your database with this data. You should include screen shots of the scripts running within the database environment.

You should provide an explanation of how you developed the scripts, order of data population, and any issues encountered with them.

**4.1 Data Population Screenshot**

**Cstomer data**

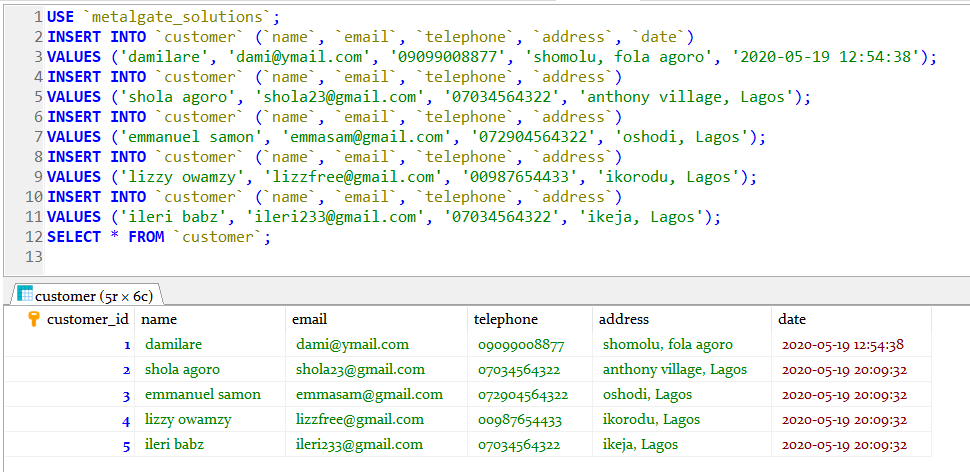
****

Fig 4.0 Screenshot of customers data population

**Department**

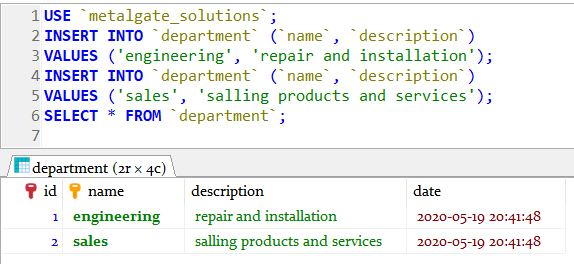
****

Fig 4.1 Screenshot of department data population

**Status**

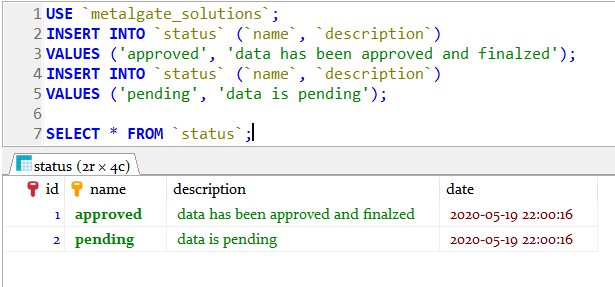
****

Fig 4.2 Screenshot of status table data population

**Employee**

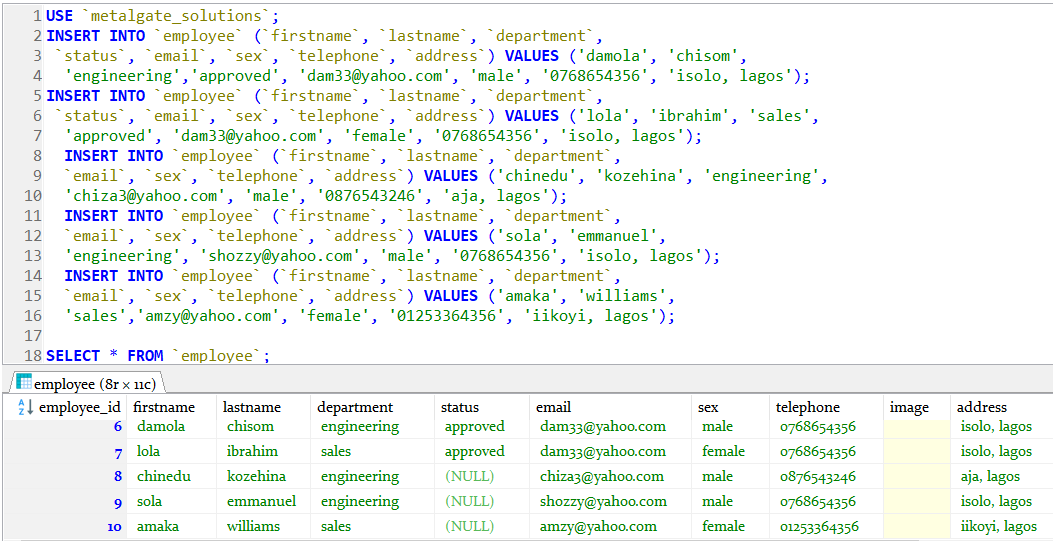
****

Fig 4.3 Screenshot of employee table data population

**Supplier**

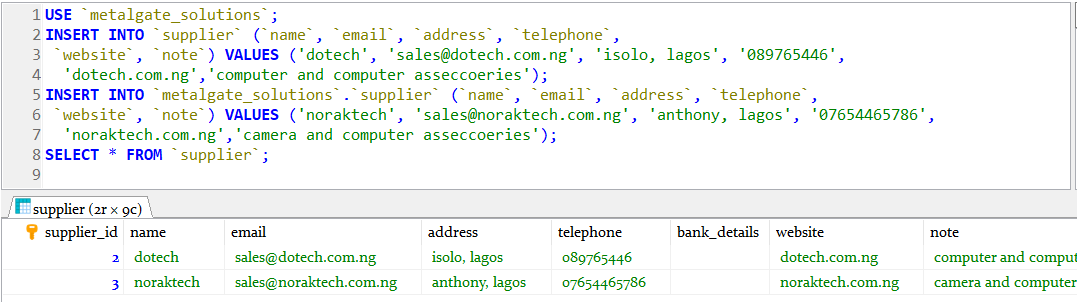
****

Fig 4.4 Screenshot of supplier table data population

**Receipt\_type**

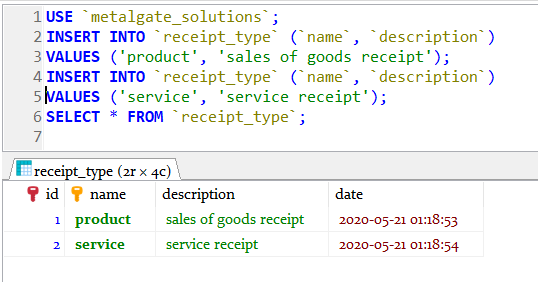
****

Fig 4.5 Screenshot of employee table data population

**Item\_category**

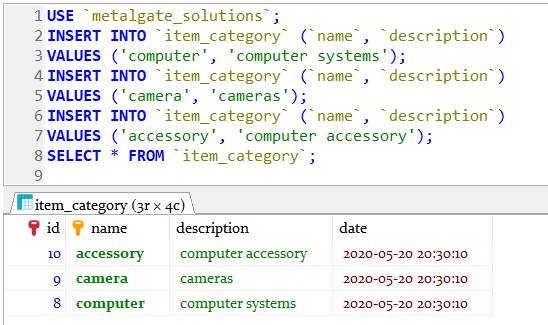
****

Fig 4.6 Screenshot of item\_category table data population

**Items**

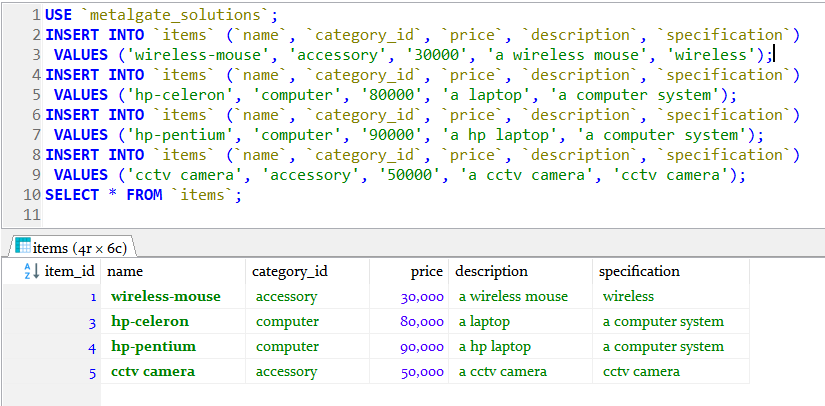
****

Fig 4.7 Screenshot of items table data population

**Purchase**

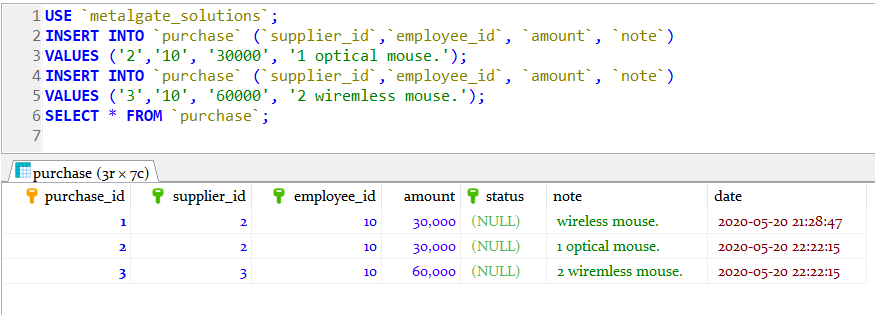
****

Fig 4.8 Screenshot of purchase table data population

**Purchase\_list**

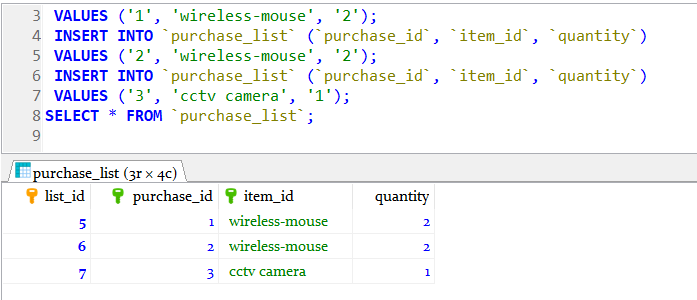
****

Fig 4.9 Screenshot of purchase\_list table data population

**Item\_update**

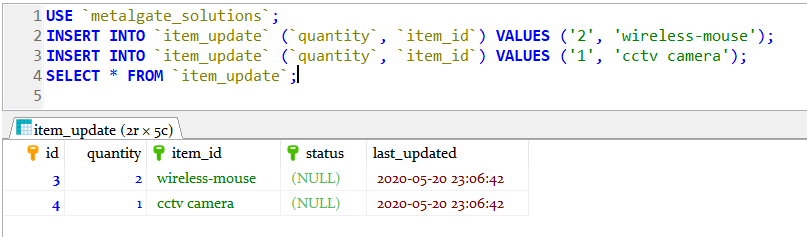
****

Fig 4.10 Screenshot of item\_update table data population

**Invoice**

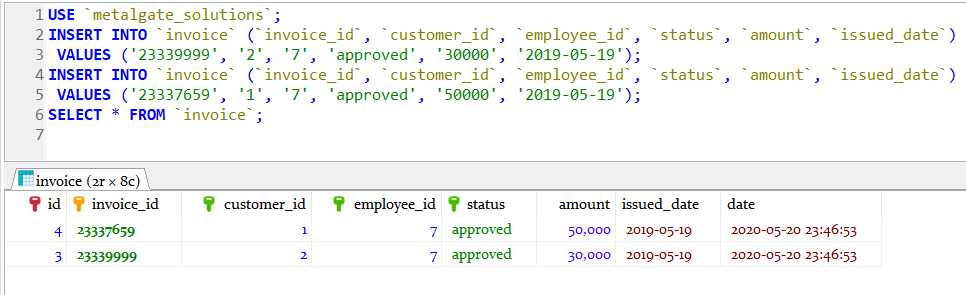
****

Fig 4.11 Screenshot of invoice table data population

**Invoice\_item**

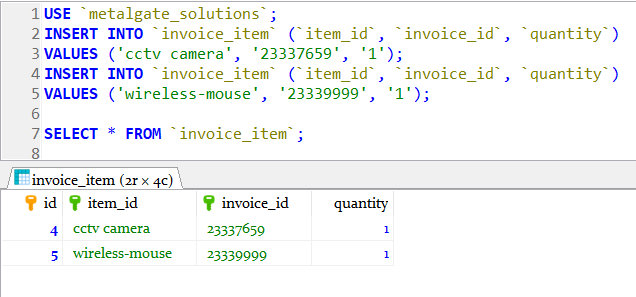
****

Fig 4.12 Screenshot of invoice\_item table data popu13ation

**Quotation**

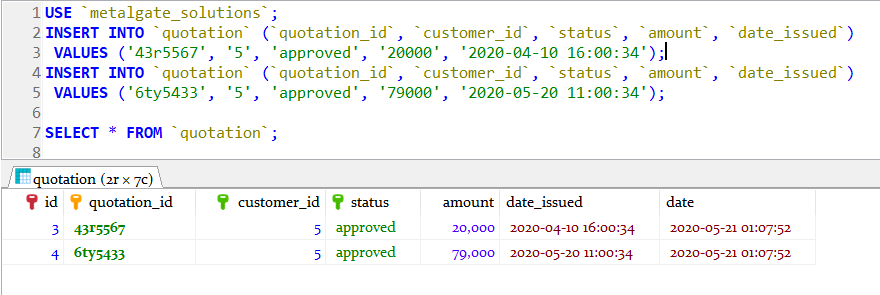
****

Fig 4.13 Screenshot of Quotation table data population

**Receipt**

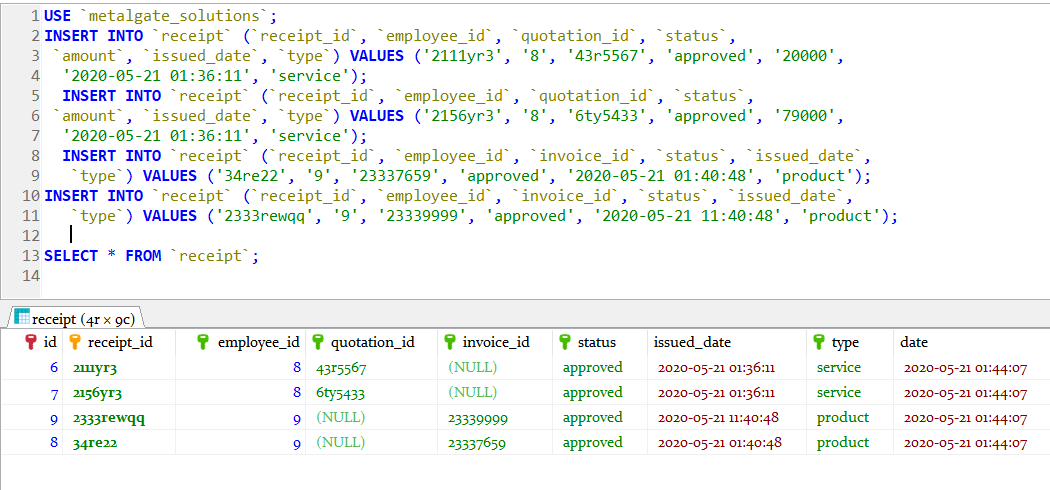
****

Fig 4.14 Screenshot of Receipt table data population

**Service\_Type**

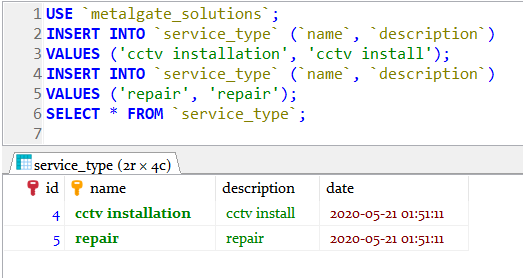
****

Fig 4.15 Screenshot of Service\_type table data population

**Service**

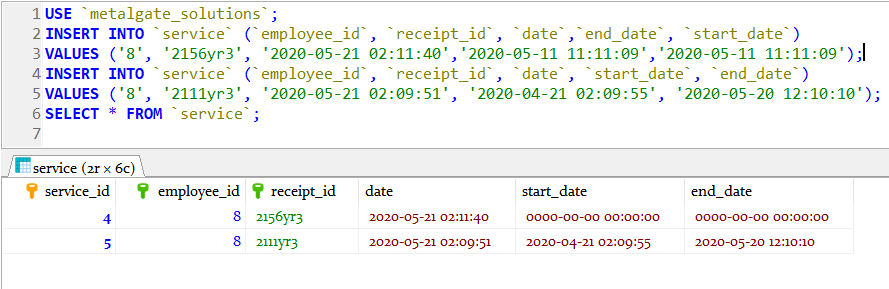
****

Fig 4.16 Screenshot of Service table data population

**4.2 Description of Data Population Script**

To prevent integrity constraint violation, I populated the 1NF tables (CUSTOMERS, DEPARTMENT, STATUS, SERVICE\_TYPE, RECEIPT\_TYPE, ITEM\_CATEGORY AND SUPPLIER). And then I populated the EMPLOYEE, ITEMS, PURCHASE and PURCHASE\_LIST table respectively. the QUOTATION, INVOICE, INVOICE\_ITEM and ITEM\_UPDATE table, then lastly, I populated the RECEIPT and SERVICE table, respectively because the service table is dependent on the RECEIPT.

**4.3 Errors Encountered**

**4.3.1 Domain and foreign key constraint error**

I had to alter some tables by setting the foreign key column default to NULL

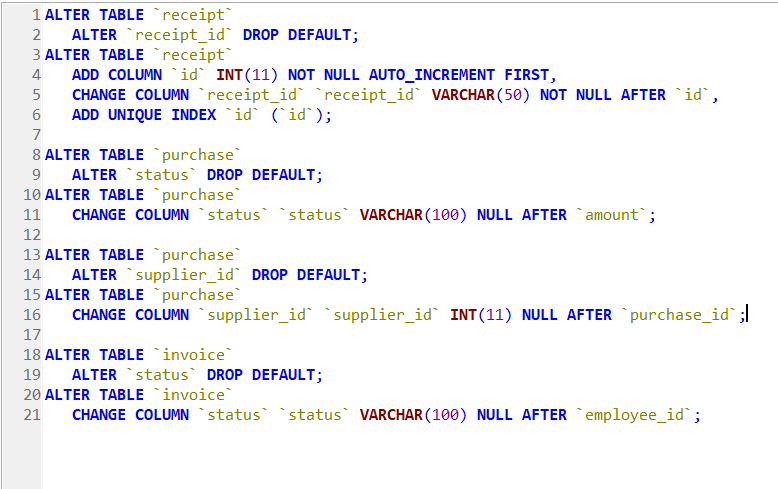
****

Fig 4.17 Screenshot of altered tables to rectify the foreign and domain contraint error during data population

**4.3.2 Duplicate Primary Key Data**

This occurred due to lack of unique key assigned to some of the primary key that are not auto incremented,. This was resolved by adding a UNIIQUE key to the entities affected.

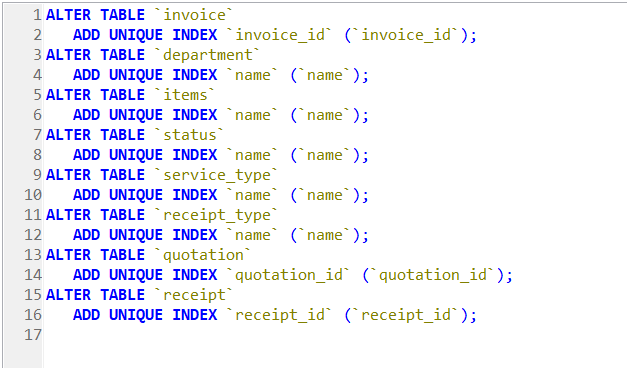


Fig 4.17 Screenshot of altered tables to rectify the duplicate primary key error during data population

**4.3.3 Missing Column**

Populating the invoice\_item entity, there was a missing column INVOICE\_ID

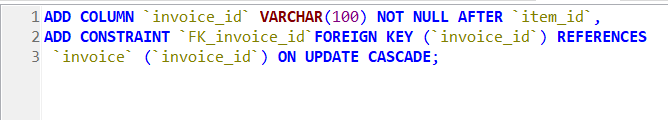


Fig 4.17 Screenshot of altered tables to rectify the missing column error encountered during data population

**5.0 Task 5 – Enhancements (15 marks)**

Discuss how you would design and implement THREE (3) enhancements to your current systems. These could include changes in scope, data requirements, functionality, or user interface design.

You should provide a rationale for each of your enhancements that demonstrates how they would be useful for your system and chosen organisation.

* 1. **Enhancements**

Change in scope, user interface design, functionality

**5.1.1 Change in Scope**

Adding more relevant entities such as JOB with relevant columns such as job\_title and salary thereby creating an automated salary structure for the organisation.

**5.12 User Interface Design**

Creating a user interface design would help the organisation automate tasks, and help reduce human error. They would be able to add data to the database and see relevant result based on their needs. This interface would be web based and designed with HTML, CSS, JAVASCRIPT and PHP.

* + 1. **Enhance the functionality**

I would create an avenue where customers can order and pay for goods and services online, this would require more entities added and also the customer entity modified. This would greatly increase the organisation revenue.

**6.0 Task 6 – SQL reports (10 marks)**

Use SQL to produce FIVE (5) queries that will be useful for your organisation. For each query you should supply the following: • Rationale for query – what is its purpose and what is it trying to retrieve • SQL script running in database environment • Result set of query shown in database environment

Query scripts and results should be shown as running in the database environment. They should be well-formatted and easy to read.

To achieve higher marks queries should be of a reasonable level of complexity. This will involve the joining of two or more tables, the use of renaming columns to ensure a user friendly result set, and the inclusion of descriptive columns from the tables.

**6.1 Useful SQL Reports**

1. **SELECT** receipt\_id, employee.firstname, employee.lastname, email, address **FROM** receipt **LEFT** **JOIN** employee **ON** receipt.employee\_id = employee.employee\_id **LEFT** **JOIN** invoice **ON** receipt.invoice\_id = invoice.invoice\_id **WHERE** invoice.customer\_id = 2 **ORDER** **BY** receipt.issued\_date **ASC** **LIMIT** 1;

The query above helps the organisation to determine the personal information of the employee that performed the last transaction of a client

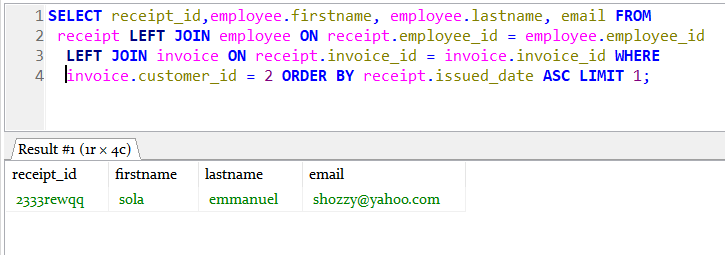


Fig 6.0 Screenshot of Query that shows the firstname lastname and email of the employee that issued a customer’s last transaction. Using Customer\_id = 2 as a case study.

1. **SELECT** **COUNT**(\*) **FROM** service **LEFT** **JOIN** employee **ON** service.employee\_id = employee.employee\_id **WHERE** employee.email = 'chiza3@yahoo.com';

The query above query helps the organization know the number of services carried out by an employee, it can be modified to determine the service rendered based on time interval

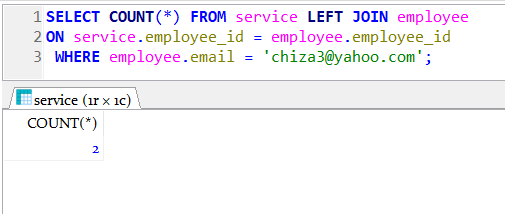
****

Fig 6.1 Query that shows number of services carried out by a particular employee, (using employee’s email as case study)

1. **SELECT** **COUNT**(\*) **FROM** receipt **LEFT** **JOIN** invoice **ON** receipt.invoice\_id = invoice.invoice\_id **LEFT** **JOIN** customer **ON** invoice.customer\_id = customer.customer\_id **WHERE** customer.email = 'shola23@gmail.com' **AND** receipt.**status** = 'approved';

The query above query helps the organization know the number of approved transactions carried out by a customer.

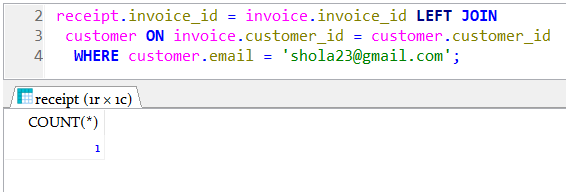
****

Fig 6.2 The number of approved transactions performed by a customer

1. **SELECT** \* **FROM** customer **WHERE** `name` **LIKE** '%z%' **OR** email **LIKE** '%la%' **OR** **date** **LIKE** '%la%' **ORDER** **BY** **date** **ASC**;

The query above enables the organisation to search for customers data based on customer name, email or date registered. the screenshot below used

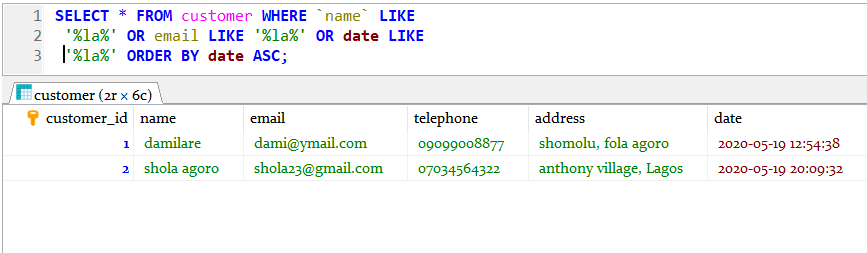
****

Fig 6.3 Search result for customers with name or email that includes “la”.

1. **SELECT** supplier.**name** **AS** Supplier\_Name, employee.firstname **AS** Employee\_firstname, employee.lastname **AS** Employee\_lastname,

employee.email, item\_id **AS** items ,purchase.date\_purchased **AS** purchased\_at **FROM** purchase **LEFT** **JOIN** purchase\_list **ON** purchase\_list.purchase\_id = purchase.purchase\_id

**LEFT** **JOIN** supplier **ON** purchase.supplier\_id = supplier.supplier\_id

**LEFT** **JOIN** employee **ON** purchase.employee\_id = employee.employee\_id **WHERE** employee.email = "amzy@yahoo.com";

The query above helps the organisation to know information of items purchased by a particular employee, date purchased, supplier name. Using employee email as identification.

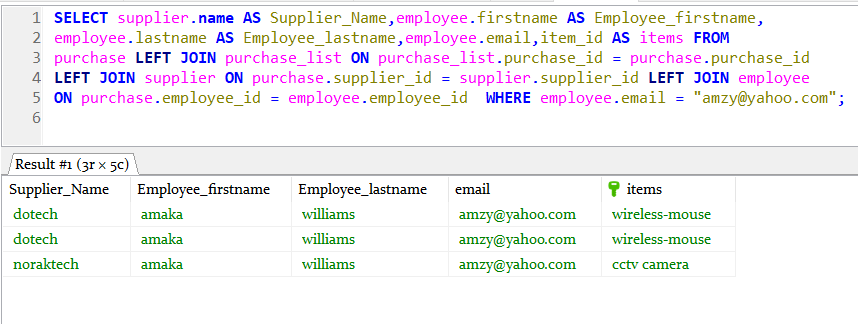


Fig 6.4 query that shows employee purchased items, supplier of the purchased items and date of the purchased item.

**7.0 Task 7 – Distributed Database Option (10 marks)**

In the future your organisation could expand by merging with other similar companies. Describe the factors that might make them consider implementing a distributed database

* 1. **Why Distributed Database**

***Reliability*:** if a failure occurs in one area of the distribution, the entire database does not experience a setback. It is better to diversify to reduce your chances of loss.

***Security*:** You can give permissions to single sections of the overall database, for better internal and external protection.

***Cost-effective*:** Bandwidth prices go down because users are accessing remote data less frequently.

***Growth*:** If you add a new location to your business, it’s simple to create an additional node within the database, making distribution highly scalable.

**8.0 Task 8 – Reflective Practice (10 marks)**

Using the Rolfe, G., Freshwater, D. and Jasper, M. (2001) model, critically review the learning that you have undertaken in order to complete this assignment.

Based upon your learning, your reflection should include a description; an analysis and; an action plan in order to bring about improvements in the future.

**8.1 Reflective Practice Using Rolfe, G (2001) Model**

**8.1.1 What?**

The assignment is to design a database system for an organisation based on their need, implementing some basic database concepts such as normalisation and integrity constraints.

**8.1.2 So what?**

However, it was not an easy task for me, because I was not so good with database design. So, I had to do a lot of research and study on database design, normalisation and the usage of constraints, which took weeks with a lot of trial and error. I had difficulties linking tables and communicating between them using joining, I had to learn that too which took quite a while.

**8.1.3 Now what?**

Proudly, I achieved a successful outcome, but to improve my database design and development skills, I am going to source for resources on Advanced Database design and development, study them rigorously and afterwards build a full-blown database to stabilize my skills.

**PAGE COUNT**: 2167

(excluding table of content, diagram texts and questions)