**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**

**“Jnana Sangama”, Belagavi-560018, Karnataka**

****

**Database Management System Lab with Mini Project**

**(18IS5DLDBL)**

REPORT

**On**

### **“Spare parts distribution Database”**

#### **BACHELOR OF ENGINEERING**

#### **In**

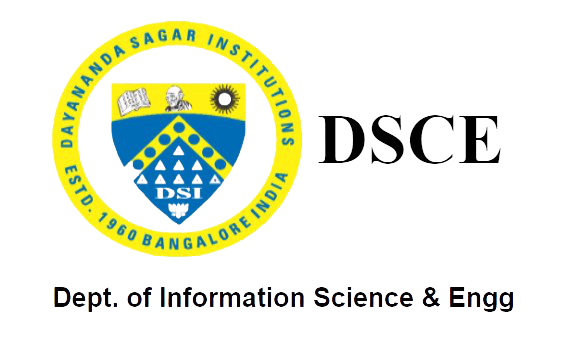
**INFORMATION SCIENCE AND ENGINEERING**

**Submitted by:**

**Vinutha J (1DS18IS086)**

#### **FACULTY IN-CHARGE:**

Mrs. Vani K A



**2020-2021**

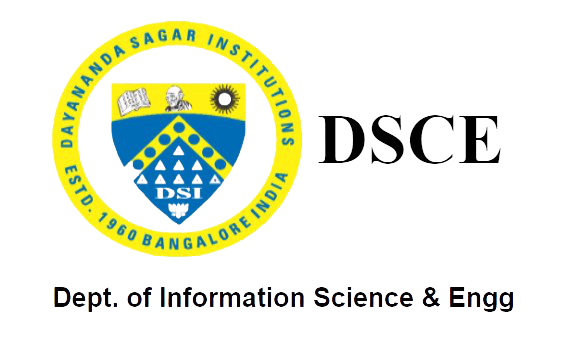
**Department Of Information Science and Engineering**

**DAYANANDA SAGAR COLLEGE OF ENGINEERING**

**SHAVIGE MALLESHWARA HILLS, KUMARASWAMY LAYOUT**

**Bengaluru-560078**

**Department of Information Science & Engineering**

****

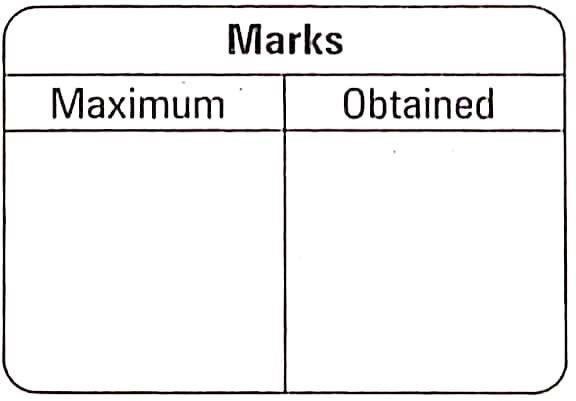
**2020-21**

**CERTIFICATE**

This is to certify that Management and Entrepreneurship, a report entitled

**“Spare parts distribution Database”** is a bonafide work carried out by

**Vinutha J** with **USN 1DS18IS086** in the partial fulfillment for the **5th semester** of Bachelor of Engineering in **Information Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the year 2020-21.



Signature of Faculty in-charge:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

[Mrs.Vani K A]

Table of Contents

1.0 INTRODUCTION

1.1 BACKGROUND

1.2 INTRODUCTION TO SPARE PARTS DATABASE SYSTEM

2.0 E R DIAGRAM AND RELATIONAL SCHEMA DIAGRAM

2.1 DESCRIPTION OF E R DIAGRAM

2.2 DESCRIPTION OF RELATIONAL SCHEMA DIAGRAM

3.0 SYSTEM DESIGN

3.1 TABLE DESCRIPTION

4.0 IMPLEMENTATION

4.1 FRONT-END DEVELOPMENT

4.2 BACK-END DEVELOPMENT

4.3 INSERTION OF A FEW RECORDS

4.4 NORMALISATION

5.0 STORED PROCEDURES

6.0 TRIGGERS

7.0 DISCUSSION OF CODE SEGMENT

Abstract

The spare parts distribution management

system is a Graphical User Interface created for the purpose of providing a platform for a distributor, to make the distribution process (managing accounts, customers, branches etc.) easy and effortless. The distributor can insert, delete and update records into the database whenever needed.

Spare parts distribution management application has a user-friendly GUI design and developed using Tkinter module in python

1.1 Background

A database is an organized collection of data, generally

stored and accessed electronically from a computer system. Where databases are more complex, they are often developed using formal design and modeling techniques.

The database management system (DBMS) is the software that interacts with end users, applications, the database itself to capture and analyze the data and provides facilities to administer the database. The sum total of the database, the DBMS and the associated applications can be referred to as a "database system". Often the term "database" is also used to loosely refer to any of the DBMS, the database system or an application associated with the database. The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified and the database schema, which defines the database’s logical structure. These three foundational elements help provide concurrency,

security, data integrity and uniform administration procedures. Typical database administration tasks supported by the DBMS include change management,

performance monitoring/tuning and backup and recovery. Many database management systems are also responsible for automated rollbacks, restarts and recovery as well as the logging and auditing of activity.

1.2 Introduction to Spare Parts Distributor Database

A spare parts database is a platform for a distributor. It provides

the distributor with all the details about a specific manufacturer and spare

parts which lets the distributor to order and sell to the customers

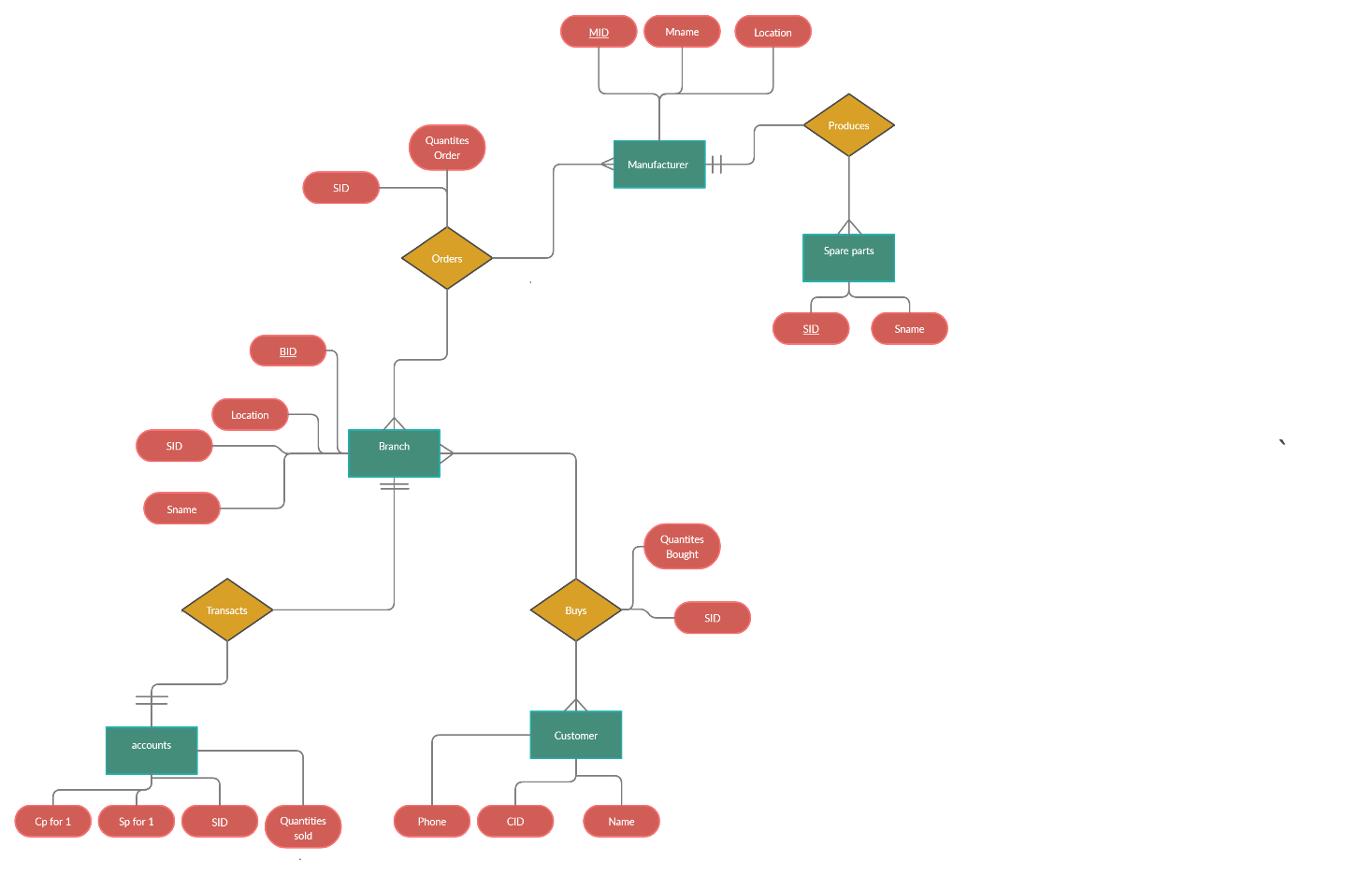
accordingly. It also helps to calculate profits based on cost price and selling

price. It is very user friendly and easy to use, making the distributor’s job

easy.

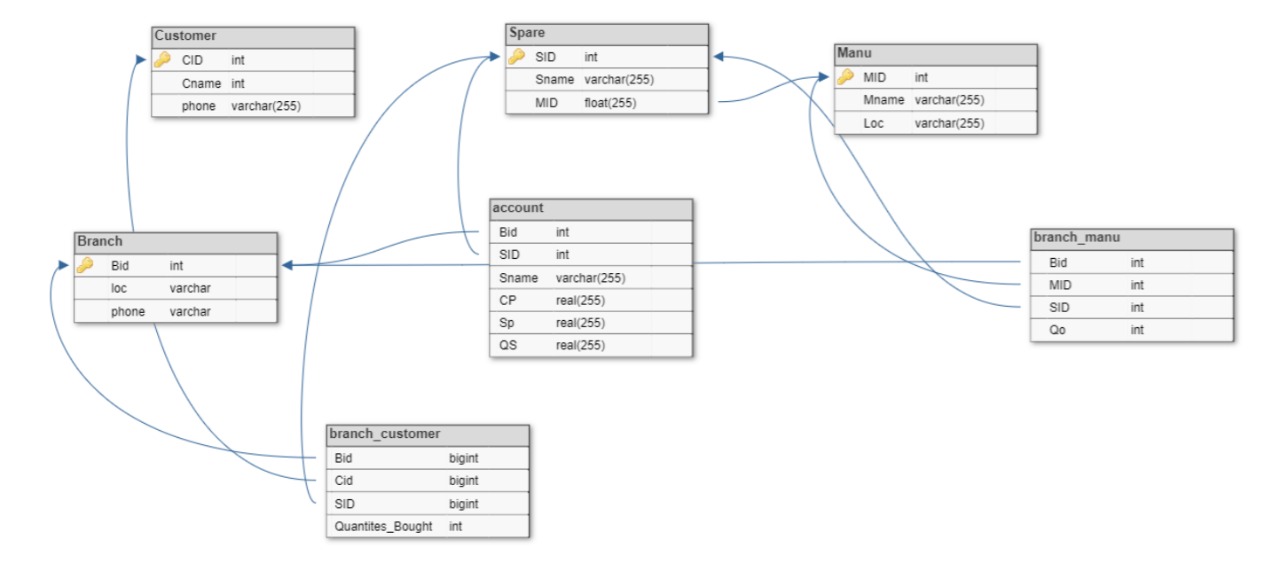
E R Diagram and Relational Schema

Description of ER Diagram



* The ER diagram describes the entities, attributes and relationships.
* Entity types like MANUFACTURER & BRANCH are in rectangular boxes.
* Relationships like are ORDERS & TRANSACTS are in diamond boxes, attached to entity types with straight lines.
* Attributes are shown in ovals, each attached by a straight line to the entity or relationship types.
* Key attributes like SID, BID and MID are underlined.
* Relational attributes are quantities\_ordered and quantities\_bought
* All the entities are in total participation in every relation.

Description of Relational Schema Diagram



The term database schema refers to the description of database that

includes the database structure and various constraints on the database.

The schema diagram is in turn an illustrative display of the database schema.

The primary keys are underlined and the referential integrity

constraints are depicted by arrows pointing to the keys they reference.

General Constraints

1. NULL Constraint: Attributes that are under NOT NULL constraints have to be filled compulsorily. Almost all the attributes in the project are under NOT NULL constraint.

2. Entity Integrity Constraint: This constraint makes sure that no

primary key can have a NULL value assigned to it. The primary keys involved in the project include:

* CID
* SID
* BID
* MID
* OID

3. Referential Integrity Constraints: A table in the back end of the

project may have references pointing to an attribute in another table. For example: BID in the BRANCH\_MANU table refers to BID in BRANCH table. The various tables are also linked with multiple foreign keys which are all set to cascade any update or delete operation on the

attribute in the main table.

The various Foreign Key attributes are:

* Spare\_MID
* Branch\_Manu\_MID
* Branch\_Manu BID
* Branch\_Cus CID
* Branch\_Cus BID

SCHEMA DESCRIPTION:

1. SPARE: This table gives us the details about all the spare parts available. It consists of Spare part ID(SID), Spare part name (Sname) and Manufacturer ID (MID).

2. BRANCH: This table contains the list of all branches of the distributor. It has the attributes as Branch ID (BID), Branch Location (loc), Phone number (phone).

3.CUSTOMER: This table gives us the details of the customer who has purchased a spare part. It includes, Customer ID (CID), Customer name (Cname) and Phone number (phone).

4. ACCOUNT: This table maintains records of the quantities sold, their cost price and selling price. We use a trigger to calculate the profit made in this table.

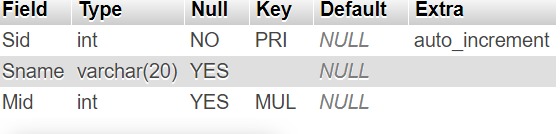
5.MANU: It contains the details of the manufacturer including the Manufacturer’s ID (MID), Name (Mname) and Location (loc).

6. Branch\_manu: This table has been derived after the ER diagram to schema conversion. It has the primary key as Order ID (OID) and BID , MID, SID as foreign keys.

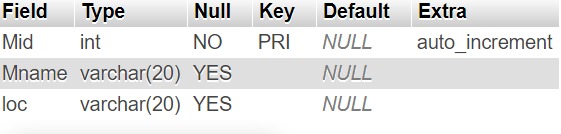
7. Branch\_cus: This table also has been derived after the ER diagram to schema conversion. It contains the Quantities bought and BID, CID, SID as the foreign keys

SYSTEM DESIGN

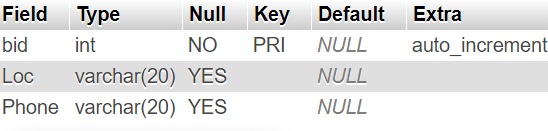
Spare:



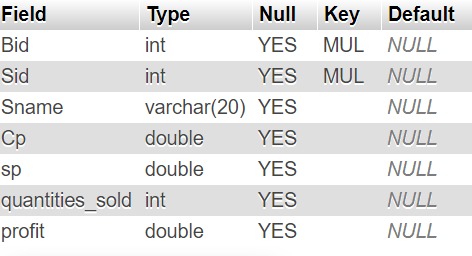
Manu:



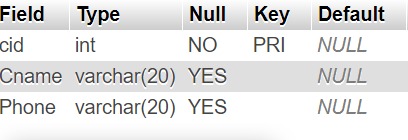
Branch:



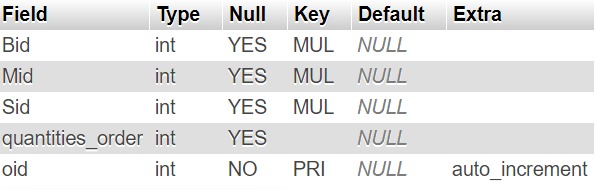
Accounts:



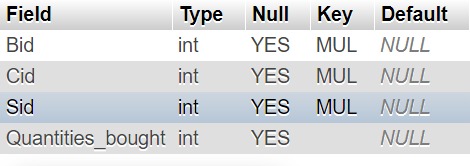
Customer:



Branch\_manu:



Branch\_customer:



IMPLEMENTATION:

Front-end Development

The front end is built using the Tkinter module in python. Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI. Tkinter is included with standard Linux, Microsoft Windows and Mac OS X installs of Python. The name Tkinter comes from Tk interface. As with most other modern Tk bindings, Tkinter is implemented as a Python wrapper around a complete Tcl interpreter embedded in the Python interpreter. Tkinter calls are translated into TCL commands, which are fed to this embedded interpreter, thus making it possible to mix Python and TCL in a single application.

Back-end Development:

It stores and arranges data, and also makes sure everything on the client- side of the website works fine. It is the part of the app that you cannot see and interact with. It is the portion of software that does not come in direct

contact with the users. The parts and characteristics developed by backend designers are indirectly accessed by users through a front-end application.

Activities, like writing APIs, creating libraries, and working with system

components without user interfaces or even systems of scientific programming, are also included in the backend.

Backend scripting language – Python

Python is a general-purpose object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Web Server -APACHE

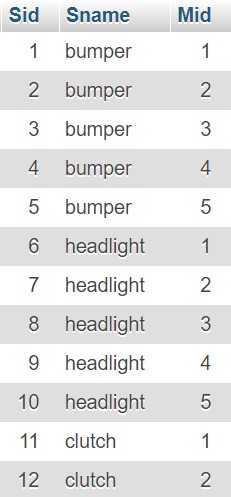
WAMP Stands for "Windows, Apache, MySQL, and PHP." WAMP is a variation of LAMP for Windows systems and is often installed as a software bundle (Apache, MySQL, and PHP). It is often used for web development and internal testing, but may also be used to serve live websites. The most important part of the WAMP package is Apache (or "Apache HTTP Server") which is used run the web server within Windows. By running a local Apache web server on a Windows machine, a web developer can test webpages in a web browser without publishing them live on the Internet. AMP also includes MySQL and PHP, which are two of the most common technologies used for creating dynamic websites. MySQL is a high-speed database, while PHP is a scripting language that can be used to access data from the database. By installing these two components locally, a developer can build and test a dynamic website before publishing it to a public web server.

Database – MySQL:

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. It is developed, marketed and supported by MySQL AB, which is a Swedish company. It is released under an open-source license. So, you have nothing to pay to use it. It is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language .It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc. It works very quickly and works well even with large data sets. It is very friendly to PHP, the most appreciated language for web development. MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB). It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

Insertion of few records in the database:

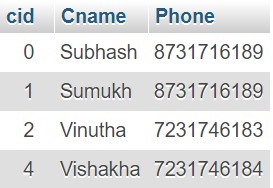
Spare:



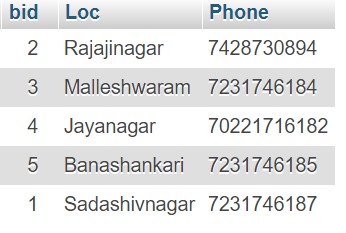
Manu:



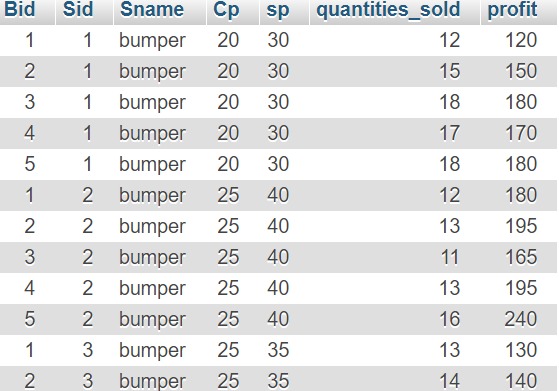
Customer:



Branch:



Account:



Branch\_manu:



Branch\_customer:



NORMALIZATION

Third Normal Form(3NF)

Third normal form (3NF) is a normal form that

is used in normalizing a

database design to reduce

the duplication of data and ensure referential

integrity by ensuring that:

1. The entity is in second normal form.

2. No non-prime (non-key) attribute is

transitively dependent on any key i.e. no non-prime attribute depends on other non-prime attributes. All the non-prime attributes must depend only on the candidate keys.

Stored Procedures

A stored procedure is a set of Structured Query Language (SQL) statements with an assigned name, which are stored in a relational database management system as a group, so it can be reused and shared by multiple programs

Create procedure manu\_spare(IN name)

SELECT s.sid, s.Sname from spare s, manu m

where m.mid = s.mid and m.Mname=name

This stored procedure will help us to determine all the spare parts which are designed by a specific manufacturer or company.

Triggers:

A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database.

Create trigger pro

before insert on

accounts

for each row

BEGIN

IF new.profit = 'yes' THEN

SET new.profit = (new.sp - new.Cp)\*new.quantities\_sold;

END IF;

END

So this trigger will automatically calculate the profit whenever a inserted into account table

Create trigger check

before insert on

branch\_manu

for each row

BEGIN

IF new.quantities\_order < 10 THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'minimum order not satisified';

END IF;

END

This trigger will make sure that minimum number while placing an order is at least 10.

**Code Review**



This code here shows the connection of app to the database and also insertion of values



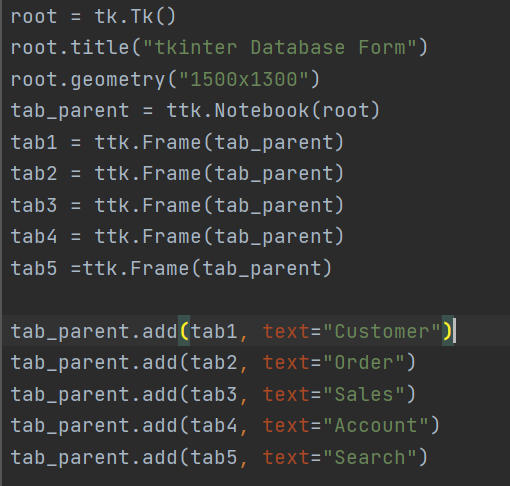
This code here shows the deletion of certain records from a table



This code shows the part where we can update certain values

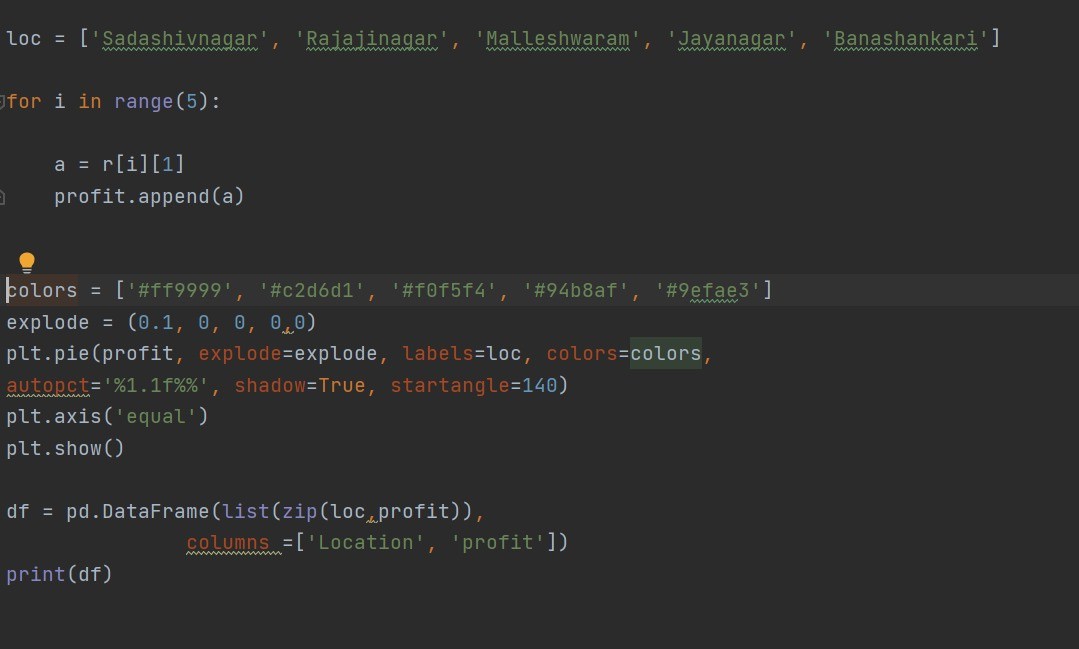


This code displays all the records of certain tables



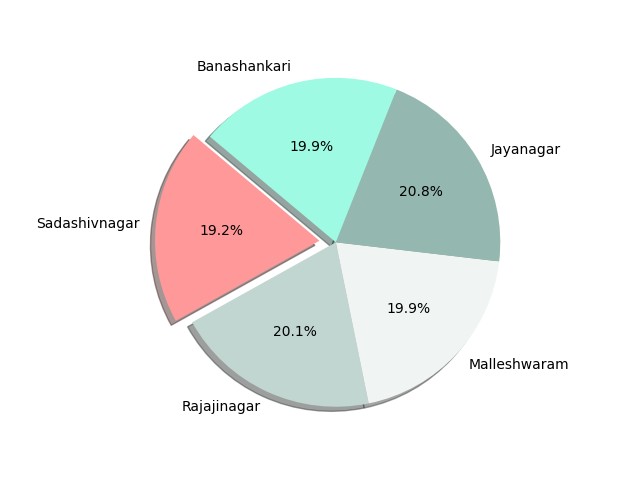
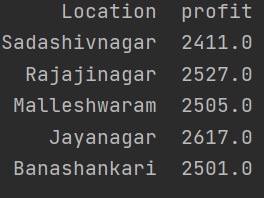
This part shows the creation a main frame and tabs of the GUI using tkinter





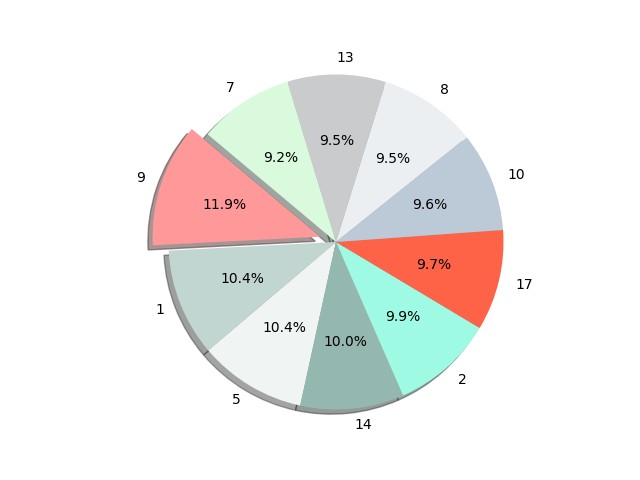
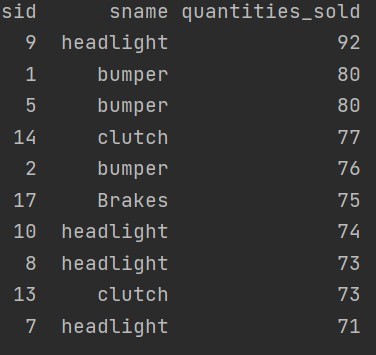
This code shows the report generation

**Report generation**



We have used matplotlib and panda to generate the tables and pie plots

The one which is shown here gives us the review of which branch earned the highest profit. As we can see it was Malleshwaram branch topping with 2505 USD



This one shows the top 10 in demand spare parts throughout all the branches and it was the headlight manufactured by Honda