Dhirubhai Ambani
Institute of Information and Communication Technology

# IT 214 DBMS
# Group Id: G4_S2.10

# Factory/Production DBMS

**Project Mentor:** Jhanavi Ma'am

| Group Member Name | Student id |
|---|---|
| Kadam Darji | 202001198 |
| Ayush H Gandhi | 202001200 |

# Index

# SECTION 1: Final Version of SRS

1. Complete Description of the Case Study/Problem domain
    A.Introduction
    B. Purpose
    C. Intended Audience and Reading Suggestions
    D. Product Scope
    E. Description
2. Fact-Finding Phase
    A.Background Reading/s
    B. Interview/s
    C. Questionnaire/s
    D. Observation/s
3. Fact-Finding Chart
4. Requirements
5. User Categories and Privileges
6. Assumptions
7. Business Constraints

# 1. Complete Description of the Problem

## Introduction:

In recent years, due to the rapid urbanization and the prevailing hot and humid climate in the region, as well as the shift in consumers' food tastes, dessert sales are constantly increasing throughout the region. According to reports, *Ice cream is one of India's most popular desserts.*

Nowadays, manual labor for processing files and document data takes a long time, especially when business procedures are not standardized. This creates a significant *bottleneck* throughout the ice cream production information scaling process. Meanwhile, with high energy consumption and pollution in manufacturing and storing ice cream in a cold environment, factories must revise the ice cream production structure in the future. Ice cream manufacturers must convert the traditional production management mode with digital and informationalizing technology to maintain an unbeatable position throughout a factory merger or consolidation.

A Factory Database Management System is a system that manages the records of factory production, storage, machines, employees, departments, buyers of the product, sale of a product, marketing statistics, and raw materials.

## Purpose:

The purpose behind creating the factory database is to build an online system that keeps track of production, profit/loss behind the production of a particular product, details of employees, buyers of that product, resources like machines, raw materials used in the production of a particular product, export statistics in different countries on a daily basis. It manages resources in one place to ease factory production management which was earlier managed and handled by humans by hiring a workforce.

Accessing information is available at your fingertips through this system instead of all overhead of data stored in files/pages. Viewing product details, updating product information or employee details, machine details, ease to make decisions about productions, and export statistics analysis is made simple and easy. These have search capabilities for finding a particular match, like employee details using the employee's

name/salary or detecting machines that need maintenance based on their working hours within a fraction of a second, together with updating any information in a table.

## Intended audience:

This project is a prototype for the ice-cream production management system of the factory, and it is ***restricted to the factory premises, employees/managers/owners, and buyers of the factory products.***

- **Factory owners:** They are the admin of the database. They have all the rights and can also add/remove managers.
- **Factory managers:** This database has been implemented under the guidance of factory managers. They can add/remove employees and see how much salary/bonus an employee deserves based on their total working days/years completed.
- **Factory employees:** They can access their details and can also claim health insurance under some conditions.
- **Production team:** They use it to make decisions regarding the production of a particular product based on the demand and stock available in the cold warehouse.
- **Sales team**: They use it to track whether products demanded are already delivered or under production. And if needed to be delivered, which vehicles among the all currently available vehicles can take demanded products. They also can use it to analyze the increase in demand of a particular product after advertising/launching new offers for that product.
- **Buyers:** They can login/register to see the details of the particular product and place an order.

## Suggested Readings:

*We learned the meaning of SRS, purpose, intended audience concepts from the resource:-* Software Requirements Specification document with example - Krazytech

*We took Introduction, Scope, and Intended Audience motivation/hints from:-* Student management system srs - Software Requirements Specification Student Management System - StuDocu

## Product Scope:

- **Digitalization and Automation** of almost all factory data.
- Accessing information is **available at your fingertips** through this system instead of all overhead of information stored in files/pages.
- The factory's Administration can use it to **modify** product/employee information at any time.
- To maintain an **unbeatable position** throughout a factory merger or consolidation by **converting the traditional production management mode *with* digital and informationalizing technology.**
- Any ice cream factory can use this system as it is **not client specific**.
- **Application support and maintenance** after deployment to production.
- Daily production and supply details, employee details with salary/bonus they deserve based on years completed, sales, profit/loss, schemes launched, machine maintenance, and export tracks can be kept using this system.

## Description:

- All these modules, in some sense, provide information about employees or the daily production and sales of the factory.
- So, all these **modules are interconnected** with each other.
  *For example*, we can calculate profit/loss from the daily sales, export statistics from buyers from different locations buying products, salary based on the department in which the employee is working and its total working days, salary/bonus they deserve, etc.
- Earlier, they were maintained manually. Therefore, all these things need to be **automated and centralized because the information from one module is needed for others.**
  *For example*, a bad working condition of a machine will lower the production of a particular week and eventually reduce profit/loss and export statistics, and vice-versa is valid for increasing machine power.

**Owner:** This relation contains information about the owner of the ice cream factory.

**Manager:** This relation has manager ID, manager name, etc. details of the managers.

**Developers:** It will have developers_id and name. The developer will have full access to change the layout/design of the database according to further requirements/challenges.

**Employees:** It will contain all the information about the employees working in the factory including their name,employee_id, address, mobile number, date of birth, qualification, age, salary, department name, joining date/since, etc.

Each employee will be working in a particular shift and under a particular department. Each employee will also have to mark its attendance while entering the factory so that based on his attendance their salary can be calculated at the end of the month. Also, employees can also claim insurance for personal or medical reasons. Employees are further divided into factory workers and drivers that drive the vehicle to deliver the orders of the ice cream products. Also, employees can be from any country having different nationalities.

Employees' details may also include previous experience of employees, origin of employees, their gender, their age when they'll retire, If they retire, their pension, if they are new to the factory, their training status.

**Pay:** This will automatically calculate the salary of the employee with their bonus to be given at the end of the particular month based on their department, leaves (more than allowed in a month if any) they had taken, bonus (if any they receive on completion of a particular tenure in the factory).

Factories always appreciate their employees by giving handsome bonuses after completion of a particular tenure in the company.

Basically, the pay will be calculated based on the attendance of the employee in that particular month and if he has completed 2/5/10 years in that month then he will receive a bonus amount. Also, the bonus amount will be full if he has not claimed any insurance but will reduce by 2% every time he has claimed the insurance. Employee can also receive the

**Department:** Department will hold the various departments of the ice cream factory, the salary of the employee working in that particular department, department_id, and bonus. Different machines will be allocated to each department and each employee will be working in one particular department. Departments also have their budget, and the number of employees in that department.

**Raw materials:** It will contain all the information on raw materials needed for particular product production, including its name, their prices. Each of them will have a unique ID. It will also contain the quantity of that raw material required for manufacturing of a particular product and also its type.

**Machines:** The machine will contain all the details regarding different machines used for the production of particular products like their name, rpm, working condition, type, and total working hours before it requires maintenance, date of last serviced month, etc. Each machine will be used in one of the departments. Each product goes through an assembly line where it is processed by different machines.
Machines' working hours keep on increasing based on the products manufactured by it as each product takes a particular time under one machine. Also, we need to keep track of working hours of the machines so that we can service them at regular intervals of time.

**Product:** It will include information about name, price, and each having a unique product ID. Each product is produced by using one or more raw_materials.
Each product will manufacture in particular quantities (possibly zero) during a particular shift on that day. It will also include its expiry date, their product code and after all count of the products that are accepted and rejected after the quality check phase. Also each product will require machines to work for a certain time interval.
After passing a quality check they will be out for processing for fssai certification. It will include information of fssai status, fssai ID, and quality check results for the product. Then, the product will be stored in the cold warehouse. Each day a product whose expiry date is closer will be carried out by staff and tries to sell at the factory outlet.

**Shift:** It will contain the shift_id, start_time, and end_time. It contains information about the shift running in the company daily with their time. Each shift contains employees working in that shift.
In each shift different products will be made from the machines allocated to it, in certain quantities that will be passed through the testing and if they passed the test then can go forward to a cold warehouse otherwise in recycling/garbage.

**Production:** It will contain all the information about the production of the particular product in a particular shift on that date i.e. machines involved in its production, date, shift, quantity, etc. After production quantities are sent for quality test and if they pass the test then they are stored in the warehouse till further orders are received for that product.

**Testing:** It contains information about the quantities of the particular product in a particular shift that cannot pass the quality test and was sent back for processing on that particular date together with the products that have passed the quality test.
The quantities which failed the test will be considered as loss of the factory in that shift.

**Cold warehouse:** It will contain the information about the products currently stored in the warehouse, which are left to deliver or not purchased with the count of each type of product. Daily production will automatically be added to the cold warehouse with the date. The sales team will use the warehouse's information to meet the buyer's demand.

**Vehicle:** It contains information like vehicle_id, driver_name, and vehicle_capacity for delivering the demanded products to its dealers. It will also store information of a particular vehicle's delivery to the buyer. It will include vehicle type whether it is truck or van, with the date it started its transport.

**Buyers:** It contains all the information about its dealers, the quantities of products they purchase, order_date, delivery_date, payment_status, delivery_vehicle_id, and at what prices the product is sold. Moreover, it will contain the information of the name of the store, address, buyer's points.

**Profit/loss:** Overall Company is working in Profit/loss? From all the above information, the system can automatically calculate and say whether the factory is running in profit or loss! Profit/loss is determined by factory_outlet, and the quantity sold of each product minus the loss for the quantities that cannot pass the quality test.

**Factory_outlet:** It will contain details of daily items sold from the factory outlet on each day with the total revenue after selling the products.

**Advertisement_impact:** It will contain the records of the impact of advertisement/launching new schemes on the sales of the particular product for which the advertisement was done. It will also include the date at which it was advertised, the details of the previous month's impact, and the current month's impact, the type of the advertisement, budget for that advertisement, and the product(s) in the ad.

**Insurance:** It is claimed by the employees. It will contain the info about the employees, insurance amount, reason of insurance, date. Each employee has the right to claim insurance under medical or personal reasons.

**Attendance:** It will contain attendance information of all employees of the factory of the month, no. of days present, no. of leaves, allowed leaves. Employees' attendance is marked by the manager.

**Factory Generators:** As factories require 24/7 electricity, generators are required when a traditional light source is not available. Therefore it will also include generators' information. It includes each generator's ID, capacity, fuel of the generator, cost of maintenance.

**Factory Security:** It will contain information about security. It will store security's name, ID, ID proof, Joining Date and the shift they are working in.

**Factory CCTV:** As a part of security, factory CCTV information is also stored together with its ID and current working status.

**Suggestion Box & Employee Complaint:** Suggestion box will accept all suggestions that anyone has and employee complaints will be open to employees to complain about the factory. This facility will be used to communicate to upper management.

**Customer Care:** To have better communication with the customer, it will have customer care service to work. Customer care will store complaints of the customers, complaints date, complaint type, product for whom the complaint is, complaint ID, customer name and any feedback the customer has.

*The **functionalities that will be available (based on the roles assigned)** in the system:-*

1) **Add_manager:** This function adds a manager to the database.

2) **Add_developer:** This function adds a developer to the database.

3) **Add_employee:** This function adds an employee to the database.

4) **Add_buyer:** This function adds a new buyer of products to the database.

5) **Omit_manager:** This function will remove a particular manager from the database.

6) **Omit_employee:** This function will be useful for removing a particular employee from the database.

7) **Omit_developer:** This function will be used for removing a particular developer.

8) **Add_raw_materials:** This function adds raw materials and their prices to the database.

9) **Add_product:** This function adds a new product to the database along with its price, raw materials, machines involved, etc.

10) **Omit_raw_materials:** This function will be useful for removing raw materials from the database.

11) **Omit_product:** This function will be useful for removing a particular product from the database that the factory no longer wants to manufacture.

12) **Calculate_Bonus:** This function will automatically see how many times the employee has claimed insurance in his tenure in the factory and will decide how much bonus they should get on his anniversary. Simply saying, Reducing the bonus amount by 2% every time he claimed the insurance that year.

13) **Add_production:** This function will automatically update the working hours of the machine and cold warehouse storage of particular products when quantities along with product details are added to the production table.

14) **Calculate_profit:** This function will automatically calculate the loss/profit when entries are made either in a testing table for quantities that failed the quality test or in the demand table when successful product delivery is made to its buyer. The buyer has made full payment or when entries are made in the factory outlet table.

# 2. Fact-Finding Phase

## Background Reading/s:

A. **Reach Manufacturing Software:**
The finest ERP for the manufacturing sector, Reach Manufacturing Software enables businesses to carry out production tasks successfully and efficiently.
- **Features:-**
  - Planning for material requirements
  - Automating requirements
  - Planning for production
  - Project management
  - Warehousing
  - Work pricing component that assists in determining the cost of each job.
  - Receive sales orders and convert them into invoices once ready

B. **Astral Manufacturing ERP (Enterprise Resource Planning):**
Astral Manufacturing ERP provides end-to-end solutions from procurement to dispatch.

- **Features:-**
    - Flexible working environment
    - Compatible with tablets, smartphones, desktops and laptops.
    - Stock management
    - Production process
    - Sale of goods
    - Dispatch & Packaging of material
    - Quality test for the material
    - Real-time tracking for batch processing concerning current status

**Flaws in both the system:**

- It doesn't include details of machine maintenance, employees working in the factory, employees insurance claimed history, employees attendance and their salary/bonus that had to be given at the end of the month, advertisement, factory outlet, transportation history.
- This software is very costly for small startup factories to afford.

**References:-**

*72 best ERP (Enterprise Resource Planning) software:-*
https://www.techjockey.com/category/manufacturing-erp?utm_id=15408797649&utm_source=Google&utm_campaign=DSA_ERP&utm_adgroup=DSA&utm_keyword=&utm_matchtype=&utm_adgroupid=133253016467&utm_creative=565220373983&utm_device=c&utm_targetid=dsa-1467081384188&utm_adposition=&gclid=CjwKCAjwp9qZBhBkEiwAsYFsby8Rz8hUuefeXTcGemAf93UrCZ1kZ8BJwCB7YzRReYNEZy-mlMsvdBoC0_sQAvD_Bw

**Combined Requirements gathered from the readings:**

- A well functioning Database management system is required to maintain and update all the information about the product, raw materials, employees, production, sales, etc.
- A user interface is required so that the employees, managers can easily access the data required without having to know much about the actual implementation of the system.
- System administrators and employees will be assigned different roles to ensure the security of the sensitive information of the factory.

- System structure and functions should be designed in such a way that it ensures the fast performance of the system.
- The interface should be clean and without any redundant data.
- System should be flexible and compatible with tablets, smartphones, desktops, and laptops.
- It should cover all the aspects of a factory, from planning for material requirements to tracking the delivery of final products, including employee details, advertisement, factory outlet, profit/loss, etc.

## Interview/s:

1. **Interview Plan (**Mock**)**

   **System:** Radhe Ice-cream Factory Database

   **Project Reference:** IT214/G4/S2/10

   **Interviewee:** 1) Arth Detroja (Role Play) **Designation:** CEO at Radhe Ice cream factory

   **Interviewer:** 1) Ayush Gandhi **Designation:** Business Development Executive- ITSolutions

   2)Kadam Darji **Designation:** Developer - ITSolutions

   **Date:** 29/9/2022 **Time:** 16:30

   **Duration:** 40 minutes **Place:** CEO's Office

   **Purpose of Interview:**

Preliminary meeting to identify problems and requirements regarding data management and to understand how the core business operates.

**Agenda:**
Problems with data management and any other concerns
Current system's procedures
Initial ideas
Follow-up actions

**Documents to be brought to the interview:**
Files having the current information of the different modules of the factory.

**Interview Summary (**Mock**)**

**System:** Radhe Ice-cream Database

**Project Reference:** IT214/G4/S2/10

**Interviewee:** 1) Arth Detroja (Role Play) **Designation:** CEO at Radhe Ice cream factory

**Interviewer:** 1) Ayush Gandhi **Designation:** Business Development Executive- ITSolutions
2) Kadam Darji **Designation:** Developer - ITSolutions

**Date:** 29/9/2022 **Time:** 16:30

**Duration:** 40 minutes **Place:** CEO's Office

**Purpose of Interview:**
Preliminary meeting to identify problems and requirements regarding data management and to understand how the core business operates.

**Summary of the Interview:**

- Business procedures are not standardized, which leads to a significant *bottleneck* throughout the ice-cream production information scaling

process.

- It is a very time-consuming task to add/update/delete details of the factory belongings.
- We must go to the particular computer to access details for even a tiny amount of information.
- As the profit/loss of a factory is determined manually, it is prone to be erroneous.
- Current system has folders for employees, machines, departments, buyers, sellers, and products storing information on each in an excel file.
- Idea is to store this information in the database. Database having relations, roles with data security for employees, manager, production team, sales team, buyers, etc., will also automate the entire production process.

## 2. Interview Plan (Mock)

**System:** Ice-cream Database

**Project Reference:** IT214/G4/S2/10

**Interviewee:** 1) Kavish Shah (Role Play) **Designation:** Product Manager at Radhe Ice cream factory

**Interviewer:** 1) Ayush Gandhi **Designation:** Business Development Executive- ITSolutions

2)Kadam Darji **Designation:** Developer - ITSolutions

**Date:** 30/9/2022 **Time:** 9:00 AM

**Duration :** 50 minutes **Place:** Kavish's Office

**Purpose of Interview:**

To determine the role of support staff and relationship to core business. To understand what resources and records are to be maintained and understand the entire production process.

**Agenda:**

Problems with current storage of information and any other concerns

Current storage procedures
Initial ideas
Follow-up actions

**Documents to be brought to the interview:**
Current details of the production for each product.

**Interview Summary (**Mock**)**

**System:** Ice-cream Database

**Project Reference:** IT214/G4/S2/10

**Interviewee:** 1) Kavish Shah (Role Play) **Designation:** Product Manager at Radhe Ice cream factory

**Interviewer:** 1) Ayush Gandhi **Designation:** Business Development Executive- ITSolutions
2)Kadam Darji **Designation:** Developer - ITSolutions

**Date:** 30/9/2022 **Time:** 9:00 AM

**Duration:** 50 minutes **Place:** Kavish's Office

**Purpose of Interview:**
To determine the role of support staff and relationship to core business. Understand what resources and records are to be maintained and the production process.
**Summary of the Interview:**
- To improve production, there isn't any common platform where all the combined information is available, starting from raw materials and machines involved in the final packaging of products.
- Change in any details of the machine/product affects the other modules, which need to be updated manually, which is time-consuming.

- Also, there are problems in calculating an employee's salary considering his department, leaves he had taken, bonus, if any, he deserves, and insurance claimed.
- Can't make decisions related to increase or decrease of particular product manufacturing as he can't figure out market demand.

**3. Interview Plan (**Mock**)**

**System:** Ice-cream Database

**Project Reference:** IT214/G4/S2/10

**Interviewee:** 1) Rishit Shah (Role Play) **Designation:** Sales Manager at Radhe Ice cream factory

**Interviewer:** 1) Ayush Gandhi **Designation:** Business Development Executive- ITSolutions

2)Kadam Darji **Designation:** Developer - ITSolutions

**Date:** 30/9/2022 **Time:** 14:30

**Duration :** 45 minutes **Place:** Rishit's Office

**Purpose of Interview :**
Preliminary meeting to identify problems and requirements regarding product's sale management at the site.

**Agenda :**
Problems with data management and any other concerns
Current system's procedures
Initial ideas
Follow-up actions

**Documents to be brought to the interview:**
Current document having information on sales of each product.

**Interview Summary (**Mock**)**

**System:** Ice-cream Database

**Project Reference:** IT214/G4/S2/10

**Interviewee:** 1) Rishit Shah (Role Play) **Designation:** Sales Manager at Radhe Ice cream factory

**Interviewer:** 1) Ayush Gandhi **Designation:** Business Development Executive- ITSolutions

2) Kadam Darji **Designation:** Developer - ITSolutions

**Date:** 30/9/2022 **Time:** 14:30

**Duration:** 45 minutes **Place:** Rishit's Office

**Purpose of Interview:**
Preliminary meeting to identify problems and requirements regarding Product sale management at the site.

**Summary of the Interview:**

- It is too hard to maintain each and every detail of buyers of the product on large scale.
- Very time taking and error-prone task to manually calculate profit/loss for each product daily.
- There is not any automation for the synchronization of the Sales team and Production team. That is, If any new product is produced the Production team has to contact the Sales team for its details and manually add it into the storage.
- Current system stores the sales team's data as an excel file.
- No feature to analyze the impact of advertisement on sales of that product.
- Need to calculate profit from the factory outlet separately.

**4. Interview Plan (**Mock**)**

**System :** Ice-cream Database

**Project Reference:** IT214/G4/S2/10

**Interviewee:** 1) Vikas Patni (Role Play) **Designation:** Employee at Radhe Ice cream factory

**Interviewer:** 1) Ayush Gandhi **Designation:** Business Development Executive- ITSolutions

2)Kadam Darji **Designation:** Developer - ITSolutions

**Date:** 1/10/2022 **Time:** 12:30

**Duration :** 20 minutes **Place:** Factory canteen

**Purpose of Interview :** To get employee feedback on the system by identifying their problems.

**Agenda :**
Problems with data management and any other concerns
Current system's procedures
Initial ideas
Follow-up actions

**Documents to be brought to the interview :**
Current document having information of the factory.

**Interview Summary (Mock)**

**System :** Ice-cream Database

**Project Reference:** IT214/G4/S2/10

**Interviewee:** 1) Vikas Patni (Role Play) **Designation:** Employee at Radhe Ice cream factory

**Interviewer:** 1) Ayush Gandhi **Designation:** Business Development Executive- ITSolutions

2) Kadam Darji **Designation:** Developer - ITSolutions

**Date:** 1/10/2022 **Time:** 12:30

**Duration :** 20 minutes **Place:** Factory canteen

**Purpose of Interview :** To get employee feedback on the system by identifying their problems.

**Summary of the Interview:**

- Employees can't validate his attendance with the system, therefore, having inconsistencies sometimes.
- No facility for employees to see their details stored with the factory.
- Current system has an excel storing the details of all the factory employees.
- Employees have to travel to the manager's office to claim insurance, leading to delays and corruption.

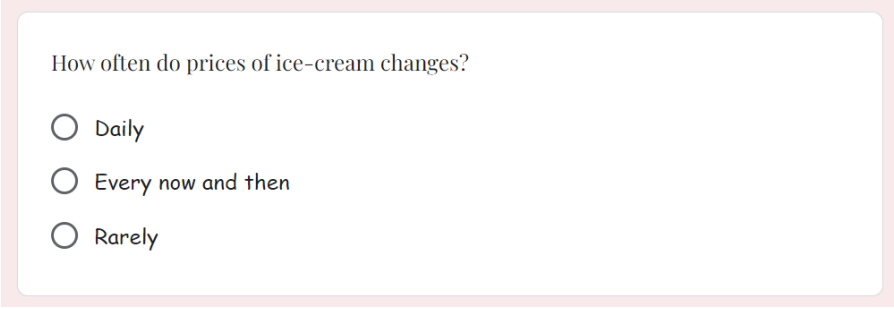
## Combined Requirements gathered from All Interviews:

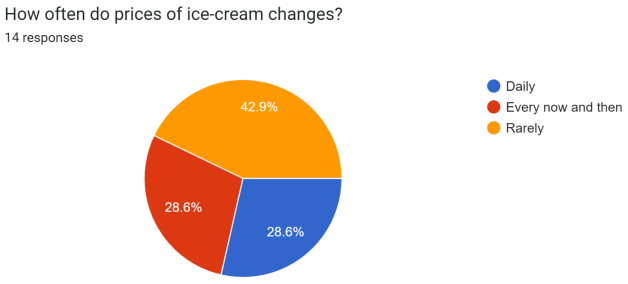
- Automate the entire process of the factory.
- System structure and functions should be designed in such a way that it ensures the fast performance of the system and integrity is maintained.
- Restriction of Data access and Data change by providing role-based security.
- Database should store the details for employees, products, departments, machines, buyers of the product, sales of the product, and profit/loss.
- Remote access to the details for employees, managers, CEO, etc.
- Common Platform across Sales Team and Production Team.
- Automate the calculation of daily profit/loss based on current production and sales.

- Employees should directly claim insurance by logging into their account and can also validate their attendance and personal details.

## **Questionnaire/s** *(Using Google form)*:


1.



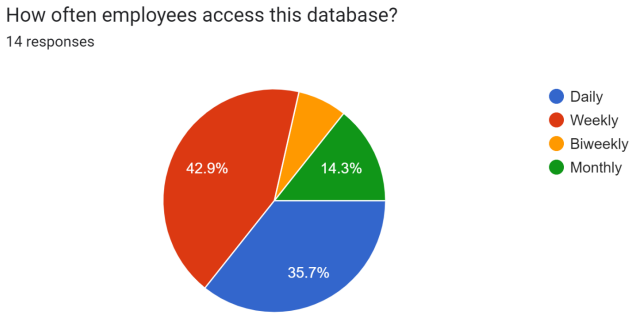




**Intent of question:** Is it necessary to create new relation for prices with date?
**Observation from the response:** No, since prices change very rarely.

2.

How often employees access this database?

- ○ Daily
- ○ Weekly
- ○ Biweekly
- ○ Monthly




**Intent of question:** Is there any need to create separate views for employees?
**Observation from the response:** Yes, as they access databases more often.

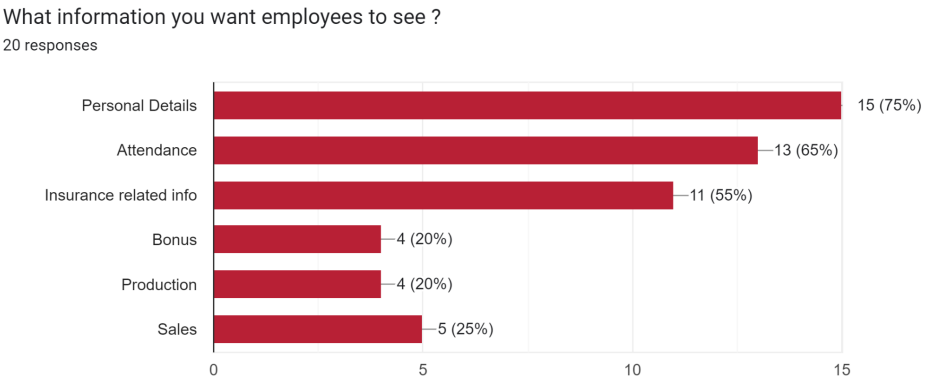
3.

What information you want employees to see ?

- ☐ Personal Details
- ☐ Attendance
- ☐ Insurance related info
- ☐ Bonus
- ☐ Production
- ☐ Sales

Any other details, you want them to see?

Your answer




**The intent of the question:** What information should be included in the employee view?
**Observation from the response:** Personal details, attendance, insurance.

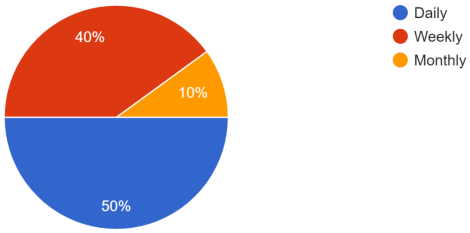
4.

How often you will access database to see production,supply,profit/loss?

- ○ Daily
- ○ Weekly
- ○ Monthly

How often you will access database to see production,supply,profit/loss?

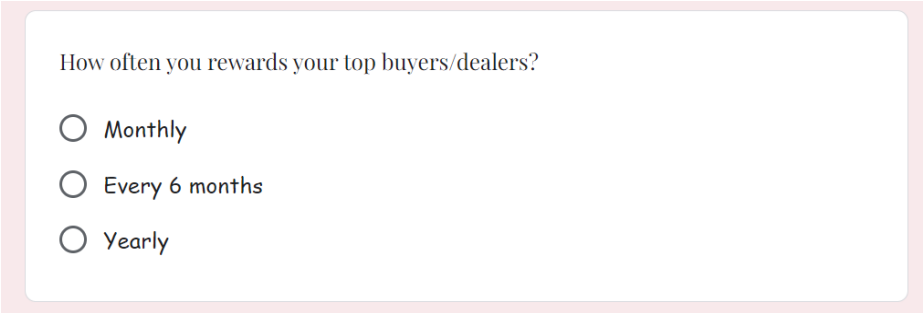
20 responses




**The intent of the question:** Should profit/loss,supply details calculate/maintain on daily basis or monthly?

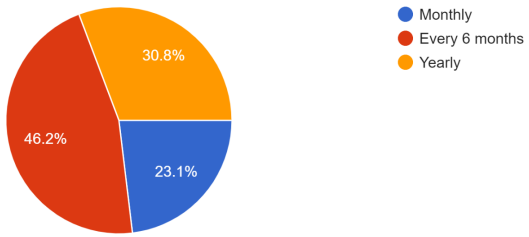
**Observation from the response:** It should be maintained/updated daily.

5.




How often you rewards your top buyers/dealers?

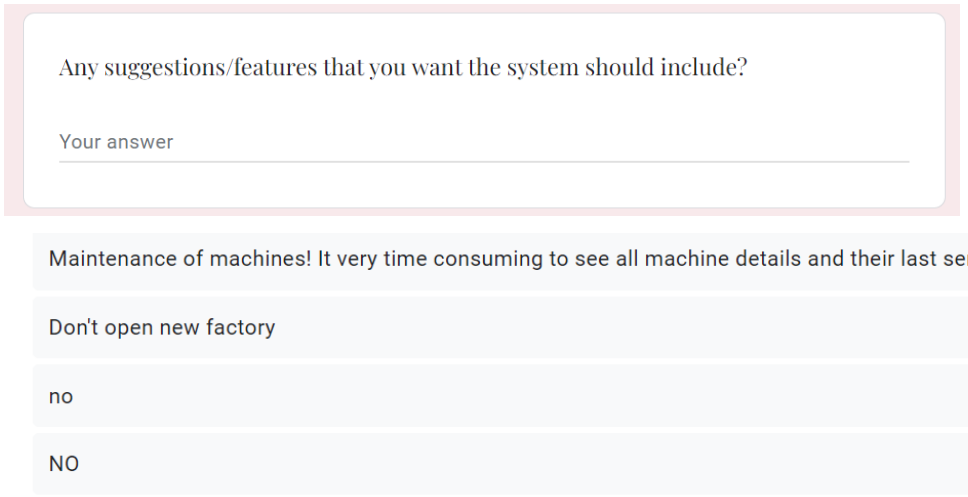
13 responses




**The intent of the question:** How often buyers list needed to be sorted?

**Observation from the response:** After every 6 months the factory rewards their top buyers every 6 months.

6.




**Intent of question:** Any other view/feature the system should implement?
**Observation from the response:** Care should be taken in arranging machines w.r.t their working hours, as most used machines will require first maintenance.

- **Combined Requirements gathered from the Questionnaire**
  - No need to create separate relations for prices of products.
  - Need to create a separate view for employees carrying personal details, attendance, insurance, etc.
  - Profit/loss  details needed to be updated every day.
  - Buyers relation should be sorted in decreasing order based on quantity of products ordered by them.
  - Machines should be sorted w.r.t their working hours, as most used machines will require first maintenance.

## Observation/s:

**ITSolutions**: Observations

**System:** Radhe Ice cream Factory

**Project Reference:** IT214/G4/S2/10

**Observations by:** Ayush Gandhi (IT Solutions- Company manager)

**Date:** 5/10/2022 **Time:** 15:10

**Duration:** 30 minutes **Place:** Radhe's Office (Radhe Ice cream factory)

**Observations:**
The **issues** with the traditional system for storing details:

1. Taking **too much time** for daily updates of the details of the factory.
2. Need to **manually update** details leading to low accuracy and prone to errors.
3. **No automation** of tasks like determining everyday profit/loss of a factory, employee salary at the end of the month, and updating machine hours based on quantities produced of a particular product.
4. Maintaining details of everything included leads to **data redundancy**.
5. **No remote access** to the details.
6. Not compatible with smartphones, laptops, tablets.
7. Privacy and security of the data were **not** given priority.
8. Potential for data loss in the system due to **system crash**.
9. **No** feature for **concurrent** accessing/updating of details.

**Combined requirements from Observations:**

The Factory Database Management System should **aim** at the following matters:

- **Automation** of day-wise production, loss/profit, and resources consumed.
- The database is updated in **real-time.**
- Different user classes should be given **different authorizations to access or update** the system.
- **Role-based security** of data should be there.
- **Providing remote access** and **compatibility** with smartphones, laptops, and tablets.
- **Assistance in making decisions** about the product based on loss/profit.

- **To manage information** on factory employees, products, departments, machines, buyers of that product, sales of that product, export statistics, etc.
- **Consistently** update changes in machines and employees.
- **Handle concurrent access** to increase performance.
- Server should be able to deal with a **large amount of data** and should be able to **manipulate data within fractions of seconds**.

# 3. Create Fact-Finding Chart

| Objective | Technique | Subject | Time commitment |
|---|---|---|---|
| To get background knowledge on factory database | Background readings | Few similar Projects and factory reports | 2 days |
| Preliminary meeting to identify problems and requirements | Interview | CEO at Radhe Ice cream factory | 40 minutes |
| To understand what resources and records are to be maintained and understanding entire production process. | Interview/ Document sampling | Product manager at Radhe Ice cream factory | 50 minutes |
| Preliminary meeting to identify requirements regarding product sales management at site | Interview | Sales manager at Radhe Ice cream factory | 45 minutes |
| To get employee feedback on the system by identifying their problems | Interview | Employee at Radhe Ice cream factory | 20 minutes |

| | | | |
|---|---|---|---|
| To get knowledge of the real world factory workflow | Observations | Manager | 1 day |
| To understand other requirements | Questionnaire | Owner of Radhe Ice cream factory | 2 days |

# 4. List Requirements

- **Automation** of day-wise production, loss/profit, and resources consumed. *(occurs at background reading & interview)*
- System structure and functions should be designed in such a way that it ensures the **fast performance** of the system. *(occurs at observation & interview)*
- System administrators and employees will be assigned **different roles** that will ensure the **security of the sensitive information** of the factory. *(occurs at background readings)*
- **Remote access** should be given and should be **compatible** with smartphones, laptops, tablets. *(occurs at background reading)*
- **Assistance in making decisions** about the particular product based on loss/profit. *(occurs at interview)*
- **To manage information** of factory employees, products, departments, machines, buyers of that product, sell of that product, export statistics, etc. *(occurs at interview)*
- **Consistently** update changes in machines and/or employees. *(occurs at interview)*
- **Handle concurrent access** inorder to increase the performance. *(occurs at background reading)*
- Server should be able to deal with **large amounts of data** and should be able to **manipulate data within fractions of seconds**. *(occurs at observations)*
- The internet and electricity should be maintained 24/7 *(occurs at background reading)*.
- Employees should **directly claim insurance** by logging into their account and can also **validate their attendance and personal details**. *(occurs at interview)*
- The interface should be **clean and without any redundant data**. *(occurs at observations)*

- **No need** to create separate relations for prices of products.*(occurs at questionnaires)*.
- Need to create a separate **view for employees** carrying personal details, attendance, insurance, etc.*(occurs at questionnaires)*
- **Profit/loss** details needed to be updated **every day**. *(occurs at questionnaires)*
- Buyers relation should be **sorted in decreasing order** based on quantity of products ordered by them.*(occurs at questionnaires)*
- Machines should be **sorted w.r.t their working hours**, as most used machine will require first maintenance.*(occurs at questionnaires)*

# 5. User Categories and Privileges

➔ **List of the user-class:**

- Owner
- Developer
- Manager
- Production team
- Sales team
- Employee

➔ **Real-World Work Flow of factory that we are planning to implement:**

- The **employees** of the factory will have a separate view in which they can see their personal details, claim insurance, check their attendance records of previous months, etc.
- The **manager** will have a separate view in which they can calculate employees salary/bonus, mark employees attendance, add/remove employees, etc.
- The **owner** of the factory will have access to almost all the relations of the factory but can only manipulate the data and not database design. He can add/remove managers, see daily profit/loss, etc.
- The **developer** will be the owner of the database and will have access to both database design (DDL) and manipulation of data entries (DML). He can change the design of the database according to the factory needs.

- **Production team** will have a separate view in which they can see stock of a particular product available in a cold warehouse and its demand and based on this they can make a decision either to increase or decrease that product manufacturing.
- **Sales team** will have a separate view in which they can see stock available in the warehouse and orders placed by the dealers which are yet to be delivered. They will also have access over advertisement impact relation so that they can analyze the impact on sales of product after advertisement/launching schemes.

# 6. Assumptions

- It is assumed that all the manager, employees, owner, developer already have their unique id and password for login which cannot be hacked.
- It is assumed that all the machines in the relation listed are in good working condition and can be manufactured at their best rate. And also they are serviced regularly before their rpm starts getting low.
- It is presumed that raw materials required daily are made available before start of the production so there is no delay.
- It is also assumed that the factory gets orders in large enough quantities that no ice-cream expires in the cold warehouse.
- It is assumed that the prices of raw materials do not change more often.
- We have assumed that the internet connection at the server is strong enough to handle all the queries from users without crashing and no loss of data occurs due to any technical failures.

# 7. Business Constraints

- There are **no branches** of the factories.
- There is **no recovery for the loss of data** at the admin side, but can be recovered when there is loss of data at the user side.
- **Every raw material should be used in production** of some (more than 0) products.
- Number of quantities produced of a particular product in a shift on a particular day **can’t be negative.**
- Employees' salary **can’t be negative and less than half** of its original salary. If employee leaves are more than attendance such that its salary gets less than half of

its original salary then that money should be deducted from next month's salary and an employee should be given at least half of its original salary.

- Employees should get a bonus on completion of his 2years/5 years/10 years tenure in the factory based on criteria decided by the factory.
- Change in month should **automatically calculate** salary of the employee, impact of advertising, reset employee attendance to 0 and store previous month attendance, production details,etc.
- All the machines **should belong** to some department but vice-versa can't say.
- **Every machine should be used in the manufacturing** of at least one product i.e. no machines should be useless.

# SECTION 2: Noun Analysis & Finalize the ER Diagram

1. Final Problem Description
2. Noun Analysis
3. Develop the ER Diagram (ERD)
4. Final Version of ERD

# 1. Final Problem Description

**Owner:** This relation contains information about the owner of the ice cream factory.

**Manager:** This relation has manager ID, manager name, etc. details of the managers.

**Developers:** It will have developers_id and name. The developer will have full access to change the layout/design of the database according to further requirements/challenges.

**Employees:** It will contain all the information about the employees working in the factory including their name,employee_id, address, mobile number, date of birth, qualification, age, salary, department name, joining date/since, etc.

Each employee will be working in a particular shift and under a particular department. Each employee will also have to mark its attendance while entering the factory so that based on his attendance their salary can be calculated at the end of the month. Also, employees can also claim insurance for personal or medical reasons. Employees are further divided into factory workers and drivers that drive the vehicle to deliver the orders of the ice cream products. Also, employees can be from any country having different nationalities.

Employees' details may also include previous experience of employees, origin of employees, their gender, their age when they'll retire, If they retire, their pension, if they are new to the factory, their training status.

**Pay:** This will automatically calculate the salary of the employee with their bonus to be given at the end of the particular month based on their department, leaves (more than allowed in a month if any) they had taken, bonus (if any they receive on completion of a particular tenure in the factory).

Factories always appreciate their employees by giving handsome bonuses after completion of a particular tenure in the company.

Basically, the pay will be calculated based on the attendance of the employee in that particular month and if he has completed 2/5/10 years in that month then he will receive a

bonus amount. Also, the bonus amount will be full if he has not claimed any insurance but will reduce by 2% every time he has claimed the insurance. Employee can also receive the

**Department:** Department will hold the various departments of the ice cream factory, the salary of the employee working in that particular department, department_id, and bonus. Different machines will be allocated to each department and each employee will be working in one particular department. Departments also have their budget, and the number of employees in that department.

**Raw materials:** It will contain all the information on raw materials needed for particular product production, including its name, their prices. Each of them will have a unique ID. It will also contain the quantity of that raw material required for manufacturing of a particular product and also its type.

**Machines:** The machine will contain all the details regarding different machines used for the production of particular products like their name, rpm, working condition, type, and total working hours before it requires maintenance, date of last serviced month, etc. Each machine will be used in one of the departments. Each product goes through an assembly line where it is processed by different machines.
Machines' working hours keep on increasing based on the products manufactured by it as each product takes a particular time under one machine. Also, we need to keep track of working hours of the machines so that we can service them at regular intervals of time.

**Product:** It will include information about name, price, and each having a unique product ID. Each product is produced by using one or more raw_materials.
Each product will manufacture in particular quantities (possibly zero) during a particular shift on that day. It will also include its expiry date, their product code and after all count of the products that are accepted and rejected after the quality check phase. Also each product will require machines to work for a certain time interval.
After passing a quality check they will be out for processing for fssai certification. It will include information of fssai status, fssai ID, and quality check results for the product. Then, the product will be stored in the cold warehouse. Each day a product whose expiry date is closer will be carried out by staff and tries to sell at the factory outlet.

**Shift:** It will contain the shift_id, start_time, and end_time. It contains information about the shift running in the company daily with their time. Each shift contains employees working in that shift.
In each shift different products will be made from the machines allocated to it, in certain quantities that will be passed through the testing and if they passed the test then can go forward to a cold warehouse otherwise in recycling/garbage.

**Production:** It will contain all the information about the production of the particular product in a particular shift on that date i.e. machines involved in its production, date, shift, quantity, etc. After production quantities are sent for quality test and if they pass the test then they are stored in the warehouse till further orders are received for that product.

**Testing:** It contains information about the quantities of the particular product in a particular shift that cannot pass the quality test and was sent back for processing on that particular date together with the products that have passed the quality test.
The quantities which failed the test will be considered as loss of the factory in that shift.

**Cold warehouse:** It will contain the information about the products currently stored in the warehouse, which are left to deliver or not purchased with the count of each type of product. Daily production will automatically be added to the cold warehouse with the date. The sales team will use the warehouse's information to meet the buyer’s demand.

**Vehicle:** It contains information like vehicle_id, driver_name, and vehicle_capacity for delivering the demanded products to its dealers. It will also store information of a particular vehicle’s delivery to the buyer. It will include vehicle type whether it is truck or van, with the date it started its transport.

**Buyers:** It contains all the information about its dealers, the quantities of products they purchase, order_date, delivery_date, payment_status, delivery_vehicle_id, and at what prices the product is sold. Moreover, it will contain the information of the name of the store, address, buyer’s points.
Buyers who have made the factory maximum profit will also get rewarded by the factory owner. The reward may be in terms of cookies, coupons, etc.

**Profit/loss:** Overall Company is working in Profit/loss? From all the above information, the system can automatically calculate and say whether the factory is running in profit or

loss! Profit/loss is determined by factory_outlet, and the quantity sold of each product minus the loss for the quantities that cannot pass the quality test.

**Factory_outlet:** It will contain details of daily items sold from the factory outlet on each day with the total revenue after selling the products.

**Advertisement_impact:** It will contain the records of the impact of advertisement/launching new schemes on the sales of the particular product for which the advertisement was done. It will also include the date at which it was advertised, the details of the previous month's impact, and the current month's impact, the type of the advertisement, budget for that advertisement, and the product(s) in the ad.

**Insurance:** It is claimed by the employees. It will contain the info about the employees, insurance amount, reason of insurance, date. Each employee has the right to claim insurance under medical or personal reasons.

**Attendance:** It will contain attendance information of all employees of the factory of the month, no. of days present, no. of leaves, allowed leaves. Employees' attendance is marked by the manager.

**Factory Generators:** As factories require 24/7 electricity, generators are required when a traditional light source is not available. Therefore it will also include generators' information. It includes each generator's ID, capacity, fuel of the generator, cost of maintenance.

**Factory Security:** It will contain information about security. It will store security's name, ID, ID proof, Joining Date and the shift they are working in.

**Factory CCTV:** As a part of security, factory CCTV information is also stored together with its ID and current working status.

**Suggestion Box & Employee Complaint:** Suggestion box will accept all suggestions that anyone has and employee complaints will be open to employees to complain about the factory. This facility will be used to communicate to upper management.

**Customer Care:** To have better communication with the customer, it will have customer care service to work. Customer care will store complaints of the customers, complaints date, complaint type, product for whom the complaint is, complaint ID, customer name and any feedback the customer has.

## Table1 - Noun Analysis

| Noun | Verb |
|---|---|
| Owner | add, remove,omit |
| Manager | marks, add, remove,omit |
| Developers | changes, is given |
| Employees | working, works(Shift), works (Department), claims (Insurance), marks, lives in(Country), gets |
| Employee ID | |
| Employee Name | |
| Employee Department | |
| Employee Salary | |
| Employee Mobile | |
| Employee Address | |
| Employee DOB | |
| **Employee Experience** | |
| **Employee Origin** | |
| **Employee Gender** | |
| **Employee Retirement Age** | |
| **Employee Pension** | |
| **Employee Training Status** | |
| **Product Expiry Date** | |
| Pay | given |
| Department | |
| Department ID | |
| Department Name | |
| Department Salary | |
| **Department Budget** | |
| Department Bonus | |
| **Department Head** | |
| **Department Size** | |
| Information | |
| **Product Code** | |
| Machine | used_in, produces |
| Machine ID | |

| | |
|---|---|
| Machine Name | |
| Machine Working Hours | |
| Machine RPM | |
| Machine Last_serviced | |
| Product | manufactured from , produced from, is carried out |
| Product ID | |
| Product Name | |
| Product Price | |
| Product Accepted Quantity | |
| Product Rejected Quantity | |
| Shift | contains |
| Shift ID | |
| Shift start time | |
| Shift end time | |
| **Product Offer** | |
| Production | contains, is done of |
| Testing | contains |
| Cold warehouse | stores |
| Vehicle | Delivers |
| Vehicles Model | |
| Vehicle Number | |
| Vehicles Status | |
| Vehicles Capacity | |
| **Vehicles Type** | |
| **Vehicles Time** | |
| Buyers | orders, from |
| **Product Packaging Size** | |
| **Product Required Environment** | |
| Insurance | paid |
| Insurance ID | |
| Insurance amount | |

| | |
|---|---|
| Insurance Date | |
| Insurance Reason | |
| **Cost** | |
| Factory_outlet Date | |
| Factory_outlet | contains |
| Factory_outlet Revenue | |
| Raw material | is used |
| Raw_material name | |
| Raw_material ID | |
| Raw_material price | |
| Raw_material quantity | |
| **Raw_material Type** | |
| **Ice cream** | |
| Factory | Makes |
| **Details** | |
| **Layout** | |
| **Database** | |
| **Requirement** | |
| **Challenges** | |
| Profit/loss | is calculated |
| **Month** | |
| **Leaves** | |
| Bonus | is calculated |
| Tenure | |
| Assembly line | |
| Advertisement | impacts |
| **System** | has |
| Company | |
| Quantity | |
| Attendance | marked |
| Attendance Month | |

| | |
|---|---|
| Attendance Present | |
| Attendance Leaves | |
| Attendance Allowed Leaves | |
| **Product Type** | |
| **Manufacturing Date** | |
| **Package Type** | |
| Country | |
| Country Name | |
| Country Currency | |
| **Factory Generators** | used |
| **Factory Generator ID** | |
| **Factory Generator Capacity** | |
| **Factory Generator Fuel** | |
| **Factory Generator Maintainenance** | |
| **Factory Security** | from |
| **Factory Security Name** | |
| **Factory Security ID** | |
| **Factory Security ID Proof** | |
| **Factory Security Joining Date** | |
| **Factory Security Shift** | |
| **Factory CCTV** | is in |
| **Factory CCTV ID** | |
| **Factory CCTV working status** | |
| **Cold warehouse product** | |
| **Cold warehouse product quantity** | |
| **Buyers Store_name** | |
| **Buyers Address** | |
| Buyers Name | |
| Buyers ID | |
| Buyers Points | |
| Buyers bill | |

| | |
|---|---|
| Buyers payment status | |
| **Buyer's order date** | |
| Buyers quantity | |
| Orders | ordered from |
| Orders Quantity | |
| Orders Payment_status | |
| Orders Date | |
| Advertisement_impact Date | |
| Advertisement_impact Previous_Month | |
| Advertisement_impact Current_Month | |
| **Advertisement_impact type** | |
| **Advertisement_impact Budget** | |
| **Advertisement_impact Product Name** | |
| **Factory Data** | holds |
| **Factory Map** | |
| **Customer Care Service** | works |
| **Customer Care Complaints** | |
| **Customer Care Complaint Date** | |
| **Customer care Complaint Type** | |
| **Customer care product type** | |
| **Customer care complaint ID** | |
| **Customer Care Customer Name** | |
| **Customer Care feedback** | |
| **Suggestion Box** | |
| **Employee Complaints** | |
| **Product fssai status** | |
| **Product fssai ID** | |
| **Product quality check** | |
| | |
| | |
| | |

| | |
|---|---|
| | |

## Table 2:

| Candidate entity set | Candidate attribute set | Candidate relationship set |
|---|---|---|
| Raw Materials | • Raw_id<br>• Name<br>• Price | is_used |
| Products | • Product_id<br>• Name<br>• Price | Produced_by, produces, profit/loss, cold_warehouse, orders&delivery, ads_impact |
| Employees | • Emp_id<br>• Name<br>• Address<br>• Mobile_number<br>• Age<br>• Date of Birth<br>• Qualifications | Includes, claims, marks, works_for, pay, drives, lives_in |
| Department | • Depart_id<br>• Name<br>• Salary<br>• Bonus | Used_in, works_for, pay |
| Shift | • Shift_id<br>• Start_time<br>• End_time | Includes, produces, profit/loss, cold_warehouse |

| | | |
|---|---|---|
| Machines | • M_id<br>• Name<br>• Working hours<br>• rpm | Used_in, produced_by, produces, profit/loss, cold_warehouse |
| Insurance | • Reason<br>• Amount<br>• Month<br>• Employee_id | Claims, pay |
| Attendance | • Month<br>• Employee_id<br>• Present<br>• Leaves<br>• Allowed_leaves | Marks, pay |
| Country | • Name<br>• Currency | lives_in |
| Factory_ outlet | • Date<br>• revenue | profit/loss |
| Vehicle | • Vehicle_id<br>• Capacity<br>• Model<br>• status | order&delivery, cold_warehouse, profit/loss, ads_impact, drives |
| Buyers | • Buyer_id<br>• Name<br>• Points<br>• Address<br>• Store_name | Ads_impact, orders&delivery, cold_warehouse, profit/loss, from |

## Table3: Rejected Nouns & Verbs

| Noun & Verbs | Rejected Reason |
|---|---|
| Owner | irrelevant |
| Manager | irrelevant |
| Developers | irrelevant |
| Employee ID | attribute |
| Employee Name | attribute |
| Employee Department | attribute |
| Employee Salary | attribute |
| Employee Mobile | attribute |
| Employee Address | attribute |
| Employee DOB | attribute |
| Employee Experience | vague |
| Employee country | attribute |
| Employee Gender | irrelevant |
| Employee Retirement Age | irrelevant |
| Employee Pension | irrelevant |
| Employee Training Status | irrelevant |
| Product Expiry Date | irrelevant |
| Pay | association |
| Department ID | attribute |
| Department Name | attribute |
| Department Salary | attribute |
| Department Budget | vague |
| Department Bonus | attribute |
| Department Head | irrelevant |
| Department Size | irrelevant |
| Information | general |
| Product Code | irrelevant |
| Machine ID | attribute |
| Machine Name | attribute |
| Machine Working Hours | attribute |
| Machine RPM | attribute |

| | |
|---|---|
| Machine Last_serviced | attribute |
| Product ID | attribute |
| Product Name | attribute |
| Product Price | attribute |
| Product Accepted Quantity | attribute |
| Product Rejected Quantity | attribute |
| Shift ID | attribute |
| Shift start time | attribute |
| Shift end time | attribute |
| Product Offer | irrelevant |
| Testing | duplicate |
| Vehicles Model | atribute |
| Vehicle Number | atribute |
| Vehicles Status | atribute |
| Vehicles Capacity | atribute |
| Vehicles Type | atribute |
| Vehicles Time | irrelevant |
| Product Packaging Size | irrelevant |
| Product Required Environment | vague |
| Insurance ID | attribute |
| Insurance amount | attribute |
| Insurance Reason | attribute |
| Factory_outlet Date | attribute |
| Factory_outlet Revenue | attribute |
| Raw_material name | attribute |
| Raw_material ID | attribute |
| Raw_material price | attribute |
| Raw_material quantity | attribute |
| Raw_material Type | vague |
| ice cream | general |
| Factory | general |

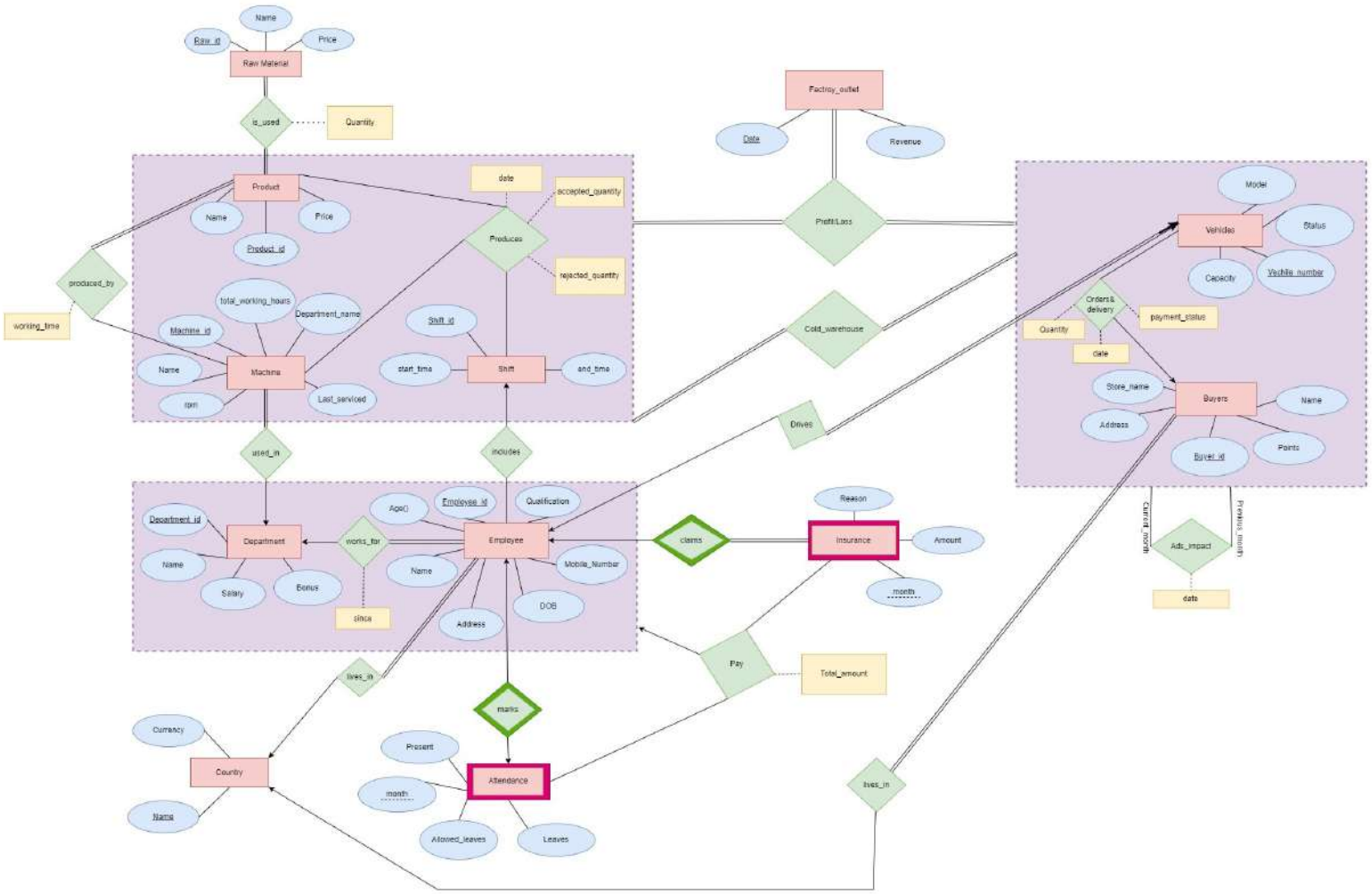
| | |
|---|---|
| Details | general |
| Layout | general |
| Database | general |
| Requirement | general |
| Challenges | general |
| Month | attribute |
| Leaves | attribute |
| Bonus | attribute |
| Tenure | general |
| Assembly line | irrelevant |
| Advertisement | irrelevant |
| System | general |
| Company | general |
| Quantity | general |
| Attendance Month | attribute |
| Attendance Present | attribute |
| Attendance Leaves | attribute |
| Attendance Allowed Leaves | attribute |
| Product Type | vague |
| Manufacturing Date | irrelevant |
| Package Type | vague |
| Country Name | attribute |
| Country Currency | attribute |
| Factory Generators | irrelevant |
| Factory Generator ID | irrelevant |
| Factory Generaor Capacity | irrelevant |
| Factory Generator Fuel | irrelevant |
| Factory Generator Maintainenance | irrelevant |
| Factory Security | duplicate (will come in employee under department security) |
| Factory Security Name | duplicate |
| Factory Security ID | duplicate |

| | |
|---|---|
| Factory Security ID Proof | duplicate |
| Factory Security Joining Date | duplicate |
| Factory Security Shift | duplicate |
| Factory CCTV | irrelevant |
| Factory CCTV ID | irrelevant |
| Factory CCTV working status | vague |
| Cold warehouse temperature | irrelevant |
| Cold warehouse product | duplicate |
| Cold warehouse product quantity | duplicate |
| Buyers Store_name | attribute |
| Buyers Address | attribute |
| Buyers Name | attribute |
| Buyers ID | attribute |
| Buyers Points | attribute |
| Buyers Store_name | attribute |
| Orders | association |
| Orders Quantity | attribute |
| Orders Payment_status | attribute |
| Orders Date | attribute |
| Advertisement_impact Date | general |
| Advertisement_impact Previous_Month | attribute |
| Advertisement_impact Current_Month | attribute |
| Advertisement_impact type | vague |
| Advertisement_impact Budget | vague |
| Advertisement_impact Product Name | attribute |
| Factory Data | general |
| Factory Map | general |
| Customer Care Service | irrelevant |
| Customer Care Complaints | irrelevant |
| Customer Care Complaint Date | irrelevant |
| Customer care Complaint Type | irrelevant |

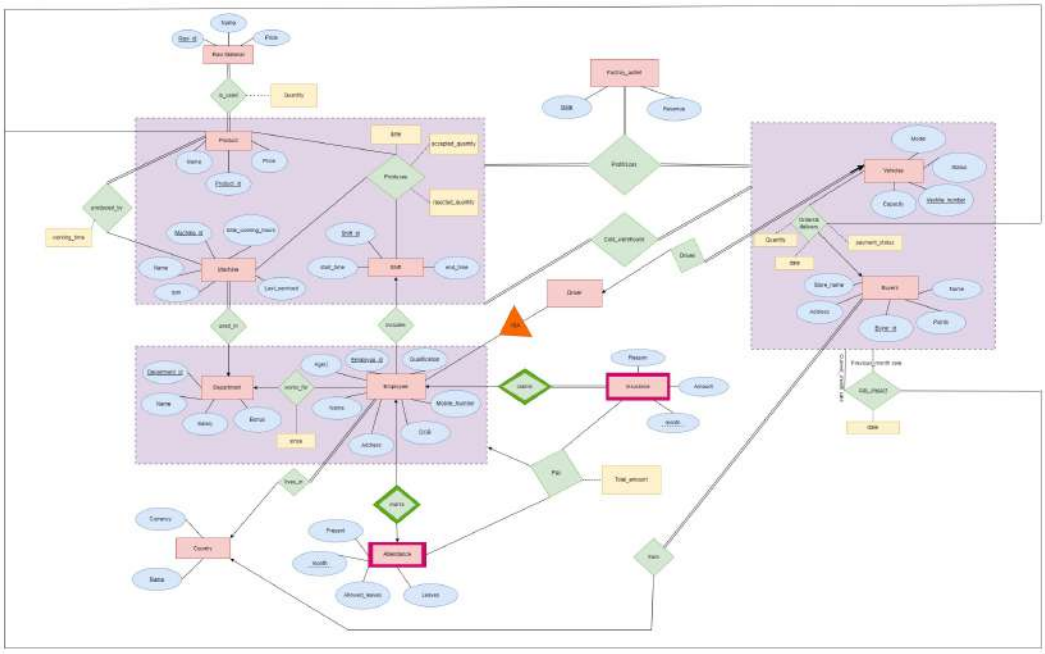
| | |
|---|---|
| Customer care product type | irrelevant |
| Customer care complaint ID | irrelevant |
| Customer Care Customer Name | irrelevant |
| Customer Care feedback | irrelevant |
| Suggestion Box | irrelevant |
| Employee Complaints | vague |
| Product fssai status | vague |
| Product fssai ID | vague |
| Product quality check | general |
| adds | irrelevant |
| include | association |
| works | association |
| is in | vague |
| claims | association |
| marks | association |
| lives | association |
| get | general |
| give | duplicate |
| holds | irrelevant |
| used | association |
| placed | duplicate |
| manufactured | duplicate |
| produced | association |
| contain | vague |
| stores | duplicate |
| paid | duplicate |
| delivers | association |
| orders | association |
| used | duplicate |
| makes | duplicate |
| calculate | irrelevant |

| | |
|---|---|
| impact | association |
| have | vague |
| marked | duplicate |
| from | duplicate |
| remove | general |
| reduce | general |
| omit | general |

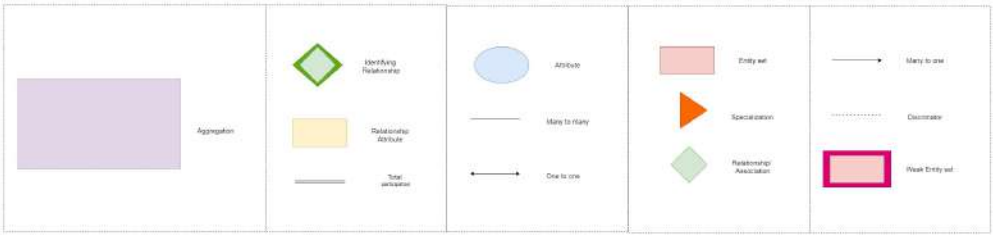
# ERD V1


Name
Raw_id
Price
Raw Material
is_used
Quantity
Product
Name
Price
Product_id
produced_by
working_time
total_working_hours
Department_name
Machine_id
Name
Machine
rpm
Last_serviced
date
accepted_quantity
Produces
rejected_quantity
Shift_id
start_time
Shift
end_time
used_in
includes
Factroy_outlet
Date
Revenue
Profit/Loss
Cold_warehouse
Drives
Model
Vehicles
Status
Capacity
Vechile_number
Orders& delivery
Quantity
payment_status
date
Store_name
Buyers
Name
Address
Buyer_id
Points
Current month
Previous month
Ads_impact
date
Department_id
Department
Name
Salary
Bonus
works_for
since
Age()
Employee_id
Qualification
Employee
Name
Mobile_Number
DOB
Address
claims
Reason
Insurance
Amount
month
Pay
Total_amount
lives_in
marks
Currency
Country
Name
Present
month
Attendance
Allowed_leaves
Leaves
lives_in

# ERD Final_Version



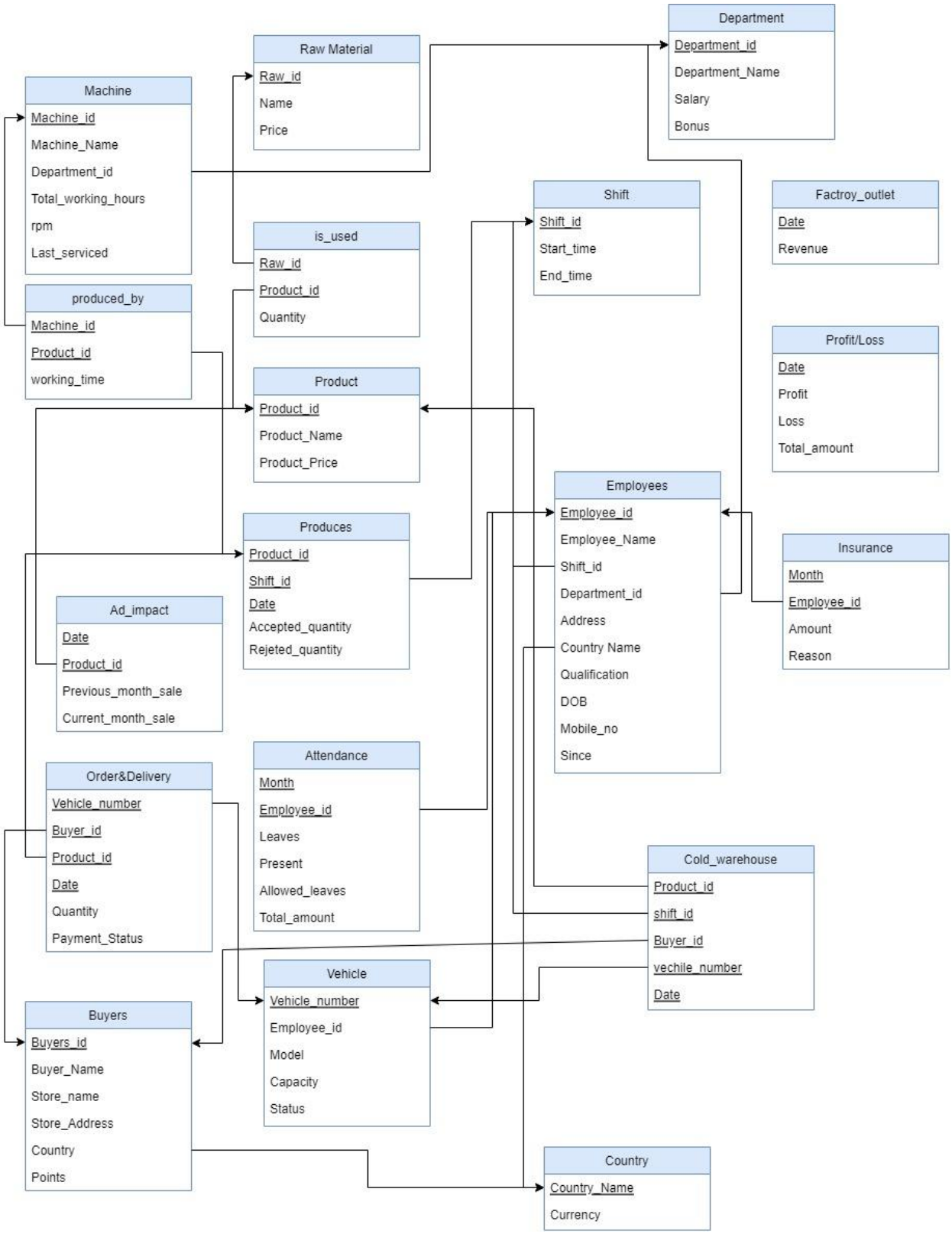
Notations:-



# Section 3: Conversion of Final ER-Diagram to Relational Model.

1. ER to Relational Mapping
2. Relational Schemas

# ER to Relational Mapping



# ER to Relational Mapping

## Relational Schemas:

Raw_materials(**Raw_id**, Name, Price)

Machine(**Machine_id**, Machine_Name, Department_id, Total_working_hours, rpm, Last_serviced)

Department(**Department_id**, Department_Name, Salary, Bonus)

Shift(**Shift_id**, Start_time, End_time)

Product(**Product_id**, Product_Name, Product_Price)

is_used(**Raw_id**, **Product_id**, Quantity)

produced_by(**Machine_id**, **Product_id**, working_time)

Produces(**Product_id**, **Shift_id**, **Date**, Accepted_quantity, Rejected_quantity)

Ad_impact(**Date**, **Product_id**, Previous_month_sale, Current_month_sale)

Employees(**Employee_id**, Employee_Name, Shift_id, Department_id, Address, Country Name, Qualification, DOB, Mobile_no, Since)

Factory_outlet(**Date**, Revenue)

Orders&Delivery(**Vehicle_number**, **Buyer_id**, **Product_id**, **Date**, Quantity, Payment_Status)

Attendance(**Month**, **Employee_id**, Leaves, Present, Allowed_leaves, Total_amount)

Insurance(**Month**, **Employee_id**, Amount, Reason)

Cold_warehouse(**Product_id**, **shift_id**, **Buyer_id**, **Vehicle_Number**, **Date**)

Buyers(**Buyers_id**, Buyer_Name, Store_Name, Store_Address, Country, Points)

Vehicle(**Vehicle_number**, Employee_id, Model, Capacity, Status)

Country(**Country_Name**, Currency)

Profit/Loss(**Date**, Profit, Loss, Total_amount)

# Description:

**Is_used**: As the relation between raw_materials and product is many to many, a separate table is made to record product with raw_material. It has the primary key as the union of raw_material_id and product_id.

**Produced_by**: The relation between machine and product is many to many, therefore this table is made to store the product with the machine. It has the primary key as the union of primary keys of machine and product table.

**Produces**: Product and Shift have a relationship many to many, therefore this table stores the product with the shift. It has the primary key as the union of both tables with the date. Date is included in the primary key to uniquely identify by date.

**Ad_impact**: It has the primary key as a union of product_id and date to get uniquely identified by particular date.

**Order&Delivery**: The relation between vehicle, buyer and product is ternary. Therefore this table stores the relevant entries of three tables and dates to uniquely identify entries with a particular date.

**Cold_warehouse**: It is a relation between two aggregates (Product, Machine, Shift & Vehicles, Buyers). It has primary key as union of Product_id, Shift_id, Buyer_id, Vehicle_Number, Date.

**Insurance**: Insurance is the weak entity therefore we have included owner's primary key employee_id to the weak entity's primary key to make it a strong entity.

**Attendance:** It is a weak entity set therefore we have included owner(employee)'s primary key to the primary key of the weak entity's primary key to make it a strong entity.

# Section 4: Normalization and Schema Refinement

## Relational Schemas (Original):

Raw_materials(**Raw_id**, Name, Price)

Machine(**Machine_id**, Machine_Name, Department_id, Total_working_hours, rpm, Last_serviced)

Department(**Department_id**, Department_Name, Salary, Bonus)

Shift(**Shift_id**, Start_time, End_time)

Product(**Product_id**, Product_Name, Product_Price)

is_used(**Raw_id**, **Product_id**, Quantity)

produced_by(**Machine_id**, **Product_id**, working_time)

Produces(**Product_id**, **Shift_id**, **Date**, Accepted_quantity, Rejected_quantity)

Ad_impact(**Date**, **Product_id**, Previous_month_sale, Current_month_sale)

Employees(**Employee_id**, Employee_Name, Shift_id, Department_id, Address, Country Name, Qualification, DOB, Mobile_no, Since)

Factory_outlet(**Date**, Revenue)

Orders&Delivery(**Vehicle_number**, **Buyer_id**, **Product_id**, **Date**, Quantity, Payment_Status)

Attendance(**Month**, **Employee_id**, Leaves, Present, Allowed_leaves, Total_amount)

Insurance(**Month**, **Employee_id**, Amount, Reason)

Cold_warehouse(**Product_id**, **shift_id**, **Buyer_id**, **Vehicle_Number**, **Date**)

Buyers(**Buyers_id**, Buyer_Name, Store_Name, Store_Address, Country, Points)

Vehicle(**Vehicle_number**, Employee_id, Model, Capacity, Status)

Country(**Country_Name**, Currency)

Profit/Loss(**Date**, Profit, Loss, Total_amount)

## Dependencies:-

**Table 1:** Raw_materials (**Raw_id**, Name, Price)

Dependencies:

- Raw_id → Name
- Raw_id → Price

**Primary Key:** Raw_id
**Foreign Key:** None
**Non-prime attributes:** Name, Price

**Anomalies:-**
Insert – None.
Update – Updating this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.
Delete – Deleting from this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 2:** Product(**Product_id**, Product_Name, Product_Price)

- Product_id → Name
- Product_id → Price

**Primary Key:** Product_id
**Foreign Key:** None
**Non-prime attributes:** Name, Price

**Anomalies:-**
Insert – None.
Update – Updating this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Delete – Deleting from this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 3:** is_used(**Raw_id**, **Product_id**, Quantity)

- Raw_id, Product_id → Quantity

**Primary Key:** Product_id,Raw_id
**Foreign Keys:** Product_id,Raw_id
**Non-prime attributes:** Quantities

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.

Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 4:** Shift(**Shift_id**, Start_time, End_time)

- Shift_id → Start_time
- Shift_id → End_time

**Primary Key:** Shift_id
**Foreign Key:** None
**Non-prime attributes:** Start_time, End_time

**Anomalies:-**
Insert – None.
Update – Updating this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.
Delete – Deleting from this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).

b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 5:** Department(**Department_id**, Department_Name, Salary, Bonus)

- Department_id → Department_Name
- Department_id → Salary
- Department_id → Bonus

**Primary Key:** Department_id
**Foreign Key:** None
**Non-prime attributes:** Salary, Bonus

**Anomalies:-**
Insert – None.
Update – Updating this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.
Delete – Deleting from this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**

**Thus, our final schema remains the same.**

**Table 6:** Factory_outlet(**Date**, Revenue)

- Date → Revenue

**Primary Key:** Date
**Foreign Key:** None
**Non-prime attributes:** Revenue

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 7:** Profit/Loss(**Date**, Profit, Loss, Total_amount)

- Date → Profit

- Date → Loss
- Date → Total_amount

**Primary Key:** Date
**Foreign Key:** None
**Non-prime attributes:** Profit, Loss, Total_amount

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 8:** Country(**Country_Name**, Currency)

- Country_name → Currency

**Primary Key:** Country_Name
**Foreign Key:** None
**Non-prime attributes:** Currency

**Anomalies:-**

Insert – None.
Update – Updating this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.
Delete – Deleting from this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 9:** Employees(**Employee_id**, Employee_Name, Shift_id, Department_id, Address, Country Name, Qualification, DOB, Mobile_no, Since)

- Employee_id → Employee_name
- Employee_id → Shift_id
- Employee_id → Departname_id
- Employee_id → Address
- Employee_id → Country_name
- Employee_id → Qualification
- Employee_id → DOB
- Employee_id → Mobile_no
- Employee_id → Since
- Mobile_Number→ Employee_id

**Primary Key:** Employee_id
**Candidate Key:** Employee_id, Mobile_no
**Foreign Key:** Shift_id, Department_id, Country_name
**Non-prime attributes:** Shift_id, Department_id, Country_name, Employee_Name, Address, DOB, Qualification, Since

**Anomalies:-**
Insert – None.
Update – Updating this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.
Delete – Deleting from this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Since, there is a multivalued attribute or composite attribute therefore it is **NOT 1NF.**
**Lossless Decomposition:**
We can split the table into
Employees(**Employee_id**, Employee_Name, Shift_id, Department_id, Address, Country Name, Qualification, DOB, Since) &
Mobile_Number(**Mobile_number, Employee_id**)

Now, both the tables are in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Now, this relation is in BCNF as well.**

**Thus, our final schema is CHANGED:**
Employees(**Employee_id**, Employee_Name, Shift_id, Department_id, Address, Country Name, Qualification, DOB, Since) & Mobile_Number(**Mobile_number, Employee_id**)

**Table 10:** Attendance(**Month**, **Employee_id**, Leaves, Present, Allowed_leaves, Total_amount)

- Month, Employee_id → Leaves
- Month, Employee_id → Present
- Month, Employee_id → Allowed_leaves
- Month, Employee_id → Total_amount

**Primary Key:** Month, Employee_id
**Foreign Key:** None
**Non-prime attributes:** Leaves, Present, Allowed_leaves, Total_amount

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**

**Thus, our final schema remains the same.**

**Table 11:** Vehicle(**Vehicle_number**, Employee_id, Model, Capacity, Status)

- Vehicle_id → Employee_id
- Vehicle_id → Model
- Vehicle_id → Capacity
- Vehicle_id → Status

**Primary Key:** Vehicle_id
**Foreign Key:** Employee_id
**Non-prime attributes:** Employee_id, Model, Capacity, Status

**Anomalies:-**
Insert – None.
Update – Updating this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.
Delete – Deleting from this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.

**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 12:** Buyers(**Buyers_id**, Buyer_Name, Store_Name, Store_Address, Country, Points)

- Buyers_id → Buyer_Name
- Buyers_id → Store_Name
- Buyers_id → Store_Address
- Buyers_id → Country
- Buyers_id → Points

**Primary Key:** Buyers_id
**Foreign Key:** Country
**Non-prime attributes:** Buyers_Name, Store_address, Store_Name, Country, Points

**Anomalies:-**
Insert – None.
Update – Updating this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.
Delete – Deleting from this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,

a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 13:** Insurance(**Month**, **Employee_id**, Amount, Reason)

- Month, Employee_id → Amount
- Month, Employee_id → Reason

**Primary Key:** Employee_id, Month
**Foreign Key:** Employee_id
**Non-prime attributes:** Amount

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 14:** Cold_warehouse(**Product_id**, **shift_id**, **Buyer_id**, **Vehicle_Number**, **Date**)

- Product_id, Buyer_id, Shift_id, Date, Vehicle_number → Product_id, Buyer_id, Shift_id, Date, Vehicle_number

**Primary Key:** Product_id, Buyer_id, Shift_id, Date, Vehicle_number
**Foreign Key:** Product_id, Buyer_id, Shift_id, Vehicle_number
**Non-prime attributes:** None

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 15:** Machine(**Machine_id**, Machine_Name, Department_id, Total_working_hours, rpm, Last_serviced)

- Machine_id → Machine_Name

- Machine_id → Department_id
- Machine_id → Total_working_hours
- Machine_id → rpm
- Machine_id → Last_serviced

**Primary Key:** Machine_id
**Foreign Key:** None
**Non-prime attributes:** Machine_Name, Department_Name, Total_working_hours, rpm, Last_serviced

**Anomalies:-**
Insert – None.
Update – Updating this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.
Delete – Deleting from this table will create problems sometimes as its primary key is being accessed as a foreign key in other tables.

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 16:** produced_by(**Machine_id**, **Product_id**, working_time)

- Machine_id, Product_id → working_time

**Primary Key:** Machine_id, Product_id
**Foreign Key:** Macine_id, Product_id
**Non-prime attributes:** working_time

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 17:** Produces(**Product_id**, **Shift_id**, **Date**, Accepted_quantity, Rejected_quantity)

- Product_id, Shift_id, Date → Accepted_quantites
- Product_id, Shift_id, Date → Rejected_quantites

**Primary Key:** Shift_id, Product_id, Date
**Foreign Key:** Shift_id, Product_id
**Non-prime attributes:** Accepted_quantities, Rejected_quantites

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 18:** Ad_impact(**Date**, **Product_id**, Previous_month_sale, Current_month_sale)

- Date, Product_id → Previous_month_sale
- Date, Product_id → Current_month_sale

**Primary Key:** Date, Product_id
**Foreign Key:** Product_id
**Non-prime attributes:** Previous_month_sale, Current_month_sale

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.
Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

**Table 19:** Orders&Delivery(**Vehicle_number**, **Buyer_id**, **Product_id**, **Date**, Quantity, Payment_Status)

- Vehicle_number, Buyer_id, Product_id, Date → Quantity
- Vehicle_number, Buyer_id, Product_id, Date → Payment_status

**Primary Key:** Vehicle_number, Buyer_id, Product_id, Date
**Foreign Key:** Vehicle_number, Buyer_id, Product_id
**Non-prime attributes:** Quantity, Payment_status

**Anomalies:-**
Insert: None
Delete: None
Update: None

Since, there is no multivalued attribute or composite attribute therefore it is in 1NF.

Also, it has no partial dependencies as it has simple PK therefore it is also in 2NF.
Since there is no transitive dependency, it is also in 3NF.
For a relation to be in BCNF,
a. It should be in the third normal form (3NF).
b. For any dependency A → B, A must be a super/candidate key.
**Hence, this relation is in BCNF as well.**
**Thus, our final schema remains the same.**

## Final relations with the schema:

1. Raw_materials(**Raw_id**, Name, Price)

2. Machine(**Machine_id**, Machine_Name, Department_id,
   Total_working_hours, rpm, Last_serviced)

3. Department(**Department_id**, Department_Name, Salary, Bonus)

4. Shift(**Shift_id**, Start_time, End_time)

5. Product(**Product_id**, Product_Name, Product_Price)

6. is_used(**Raw_id**, **Product_id**, Quantity)

7. produced_by(**Machine_id**, **Product_id**, working_time)

8. Produces(**Product_id**, **Shift_id**, **Date**, Accepted_quantity,
   Rejected_quantity)

9. Ad_impact(**Date**, **Product_id**, Previous_month_sale, Current_month_sale)

10. Employees(**Employee_id**, Employee_Name, Shift_id, Department_id, Address, Country Name, Qualification, DOB, Since)

11. Mobile_Number(**Mobile_no, Employee_id**)

12. Factory_outlet(**Date**, Revenue)

13. Orders&Delivery(**Vehicle_number**, **Buyer_id**, **Product_id**, **Date**, Quantity, Payment_Status)

14. Attendance(**Month**, **Employee_id**, Leaves, Present, Allowed_leaves, Total_amount)

15. Insurance(**Month**, **Employee_id**, Amount, Reason)

16. Cold_warehouse(**Product_id**, **shift_id**, **Buyer_id**, **Vehicle_Number**, **Date**)

17. Buyers(**Buyers_id**, Buyer_Name, Store_Name, Store_Address, Country, Points)

18. Vehicle(**Vehicle_number**, Employee_id, Model, Capacity, Status)

19. Country(**Country_Name**, Currency)

20. Profit/Loss(**Date**, Profit, Loss, Total_amount)

# Section 5: SQL

1. SQL: Final DDL Scripts
2. Insert statements
3. 20 SQL Queries with Snapshots
4. Function
5. Trigger Function

# Final DDL Scripts

```
SET search_path to public;

-----------------------DDL Scripts in public Schemas---------------------

CREATE TABLE IF NOT EXISTS "Raw_materials"
(
    "Raw_id" integer NOT NULL,
    "Name" character varying COLLATE pg_catalog."default" NOT NULL,
    "Price" smallint NOT NULL,
    CONSTRAINT "Raw_materials_pkey" PRIMARY KEY ("Raw_id"),
    CONSTRAINT price_check CHECK ("Price" > 0)
)

CREATE TABLE IF NOT EXISTS "Products"
(
    "Product_id" integer NOT NULL,
    "Product_name" character varying COLLATE pg_catalog."default" NOT
NULL,
    "Product_price" integer NOT NULL,
    CONSTRAINT "Products_pkey" PRIMARY KEY ("Product_id"),
    CONSTRAINT product_price_check CHECK ("Product_price" > 0)
)

CREATE TABLE IF NOT EXISTS is_used
(
    "Raw_id" integer NOT NULL,
    product_id integer NOT NULL,
    "Quantity" integer NOT NULL,
    CONSTRAINT is_used_pkey PRIMARY KEY ("Raw_id", product_id),
    CONSTRAINT "Raw_id" FOREIGN KEY ("Raw_id")
        REFERENCES "Raw_materials" ("Raw_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT product_id FOREIGN KEY (product_id)
        REFERENCES "Products" ("Product_id") MATCH SIMPLE
        ON UPDATE CASCADE
```

```
        ON DELETE CASCADE
)

CREATE TABLE IF NOT EXISTS "Shift"
(
    "Shift_id" smallint NOT NULL,
    "Start_time" time with time zone NOT NULL,
    "End_time" time with time zone NOT NULL,
    CONSTRAINT "Shift_pkey" PRIMARY KEY ("Shift_id"),
    CONSTRAINT shifts CHECK ("Shift_id" = ANY (ARRAY[1, 2, 3, 4, 5]))
)

CREATE TABLE IF NOT EXISTS "Department"
(
    "Department_id" integer NOT NULL,
    "Department_Name" character varying COLLATE pg_catalog."default" NOT
NULL,
    "Salary" integer NOT NULL,
    "Bonus" integer DEFAULT 0,
    CONSTRAINT "Department_pkey" PRIMARY KEY ("Department_id"),
    CONSTRAINT salary CHECK ("Salary" > 0)
)

CREATE TABLE IF NOT EXISTS "Factory_outlet"
(
    "Date" date NOT NULL,
    "Revenue" integer DEFAULT 0,
    CONSTRAINT "Factory_outlet_pkey" PRIMARY KEY ("Date")
)

CREATE TABLE IF NOT EXISTS "Profit/Loss"
(
    "Date" date NOT NULL,
    "Profit" integer DEFAULT 0,
    "Loss" integer DEFAULT 0,
    "Total_amount" integer DEFAULT 0,
    CONSTRAINT "Profit/Loss_pkey" PRIMARY KEY ("Date")
)

CREATE TABLE IF NOT EXISTS "Country"
```

```
(
    "Country_Name" character varying COLLATE pg_catalog."default" NOT
NULL,
    "Currency" character varying COLLATE pg_catalog."default",
    CONSTRAINT "Country_pkey" PRIMARY KEY ("Country_Name")
)

CREATE TABLE IF NOT EXISTS "Employee"
(
    "Employee_id" integer NOT NULL,
    "Employee_Name" character varying COLLATE pg_catalog."default",
    "Shift_id" integer NOT NULL,
    "Department_id" integer NOT NULL,
    "Address" character varying COLLATE pg_catalog."default",
    "Country_name" character varying COLLATE pg_catalog."default",
    "Qualification" character varying COLLATE pg_catalog."default",
    "DOB" date NOT NULL,
    "Since" character varying NOT NULL,
    CONSTRAINT "Employee_pkey" PRIMARY KEY ("Employee_id"),
    CONSTRAINT "Country" FOREIGN KEY ("Country_name")
        REFERENCES "Country" ("Country_Name") MATCH SIMPLE
        ON UPDATE SET NULL
        ON DELETE SET NULL,
    CONSTRAINT depart FOREIGN KEY ("Department_id")
        REFERENCES "Department" ("Department_id") MATCH SIMPLE
        ON UPDATE SET NULL
        ON DELETE SET NULL,
    CONSTRAINT shifts FOREIGN KEY ("Shift_id")
        REFERENCES "Shift" ("Shift_id") MATCH SIMPLE
        ON UPDATE SET NULL
        ON DELETE SET NULL
)

CREATE TABLE IF NOT EXISTS "Attendance"
(
    "Employee_id" integer NOT NULL,
    "Month" character varying NOT NULL,
    "Leaves" smallint DEFAULT 0,
    "Present" smallint DEFAULT 0,
    "Allowed_leaves" smallint DEFAULT 3,
```

```
    "Total_amount" integer DEFAULT 0,
    CONSTRAINT "Attendence_pkey" PRIMARY KEY ("Employee_id", "Month"),
    CONSTRAINT employees FOREIGN KEY ("Employee_id")
        REFERENCES "Employee" ("Employee_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)

CREATE TABLE IF NOT EXISTS "Vehicle"
(
    "Vehicle_id" integer NOT NULL,
    "Employee_id" integer NOT NULL,
    "Model" character varying COLLATE pg_catalog."default",
    "Capacity" integer NOT NULL,
    "Status" character varying COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Vehicle_pkey" PRIMARY KEY ("Vehicle_id"),
    CONSTRAINT driver FOREIGN KEY ("Employee_id")
        REFERENCES "Employee" ("Employee_id") MATCH SIMPLE
        ON UPDATE SET NULL
        ON DELETE SET NULL,
    CONSTRAINT status_check CHECK ("Status"::text = ANY
(ARRAY['available'::character varying, 'busy'::character varying,
'maintainence'::character varying]::text[]))
)

CREATE TABLE IF NOT EXISTS "Buyers"
(
    "Buyers_id" integer NOT NULL,
    "Buyer_name" character varying COLLATE pg_catalog."default",
    "Store_name" character varying COLLATE pg_catalog."default" NOT NULL,
    "Store_address" character varying COLLATE pg_catalog."default" NOT
NULL,
    "Country_name" character varying COLLATE pg_catalog."default" NOT
NULL,
    "Points" integer DEFAULT 0,
    CONSTRAINT "Buyers_pkey" PRIMARY KEY ("Buyers_id"),
    CONSTRAINT lives_in FOREIGN KEY ("Country_name")
        REFERENCES "Country" ("Country_Name") MATCH SIMPLE
        ON UPDATE SET NULL
        ON DELETE SET NULL,
```

```
    CONSTRAINT non_negative CHECK ("Points" > 0)
)

CREATE TABLE IF NOT EXISTS "Insurance"
(
    "Month" character varying NOT NULL,
    "Employee_id" integer NOT NULL,
    "Amount" bigint NOT NULL DEFAULT 0,
    "Reason" character varying COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Insurance_pkey" PRIMARY KEY ("Month", "Employee_id"),
    CONSTRAINT employee FOREIGN KEY ("Employee_id")
        REFERENCES "Employee" ("Employee_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)

CREATE TABLE IF NOT EXISTS "Cold_warehouse"
(
    date date NOT NULL,
    "Product_id" integer NOT NULL,
    shift_id integer NOT NULL,
    buyer_id integer NOT NULL,
    vehicle_number integer NOT NULL,
    CONSTRAINT "Cold_warehouse_pkey" PRIMARY KEY (date, "Product_id",
shift_id, buyer_id, vehicle_number),
    CONSTRAINT "Buyer" FOREIGN KEY (buyer_id)
        REFERENCES "Buyers" ("Buyers_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT product FOREIGN KEY ("Product_id")
        REFERENCES "Products" ("Product_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT shift FOREIGN KEY (shift_id)
        REFERENCES "Shift" ("Shift_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT vehicle FOREIGN KEY (vehicle_number)
        REFERENCES "Vehicle" ("Vehicle_id") MATCH SIMPLE
        ON UPDATE CASCADE
```

```
        ON DELETE CASCADE
)

CREATE TABLE IF NOT EXISTS "Machine"
(
    "Machine_id" integer NOT NULL,
    "Machine_name" character varying COLLATE pg_catalog."default",
    department_id integer NOT NULL,
    rpm integer NOT NULL,
    total_workig_hours integer NOT NULL,
    "Last_serviced" date NOT NULL,
    CONSTRAINT "Machine_pkey" PRIMARY KEY ("Machine_id"),
    CONSTRAINT dep FOREIGN KEY (department_id)
        REFERENCES "Department" ("Department_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)

CREATE TABLE IF NOT EXISTS produced_by
(
    product_id integer NOT NULL,
    machine_id integer NOT NULL,
    working_time smallint NOT NULL,
    CONSTRAINT produced_by_pkey PRIMARY KEY (product_id, machine_id),
    CONSTRAINT mac FOREIGN KEY (machine_id)
        REFERENCES "Machine" ("Machine_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT prod FOREIGN KEY (product_id)
        REFERENCES "Products" ("Product_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)

CREATE TABLE IF NOT EXISTS produces
(
    product_id integer NOT NULL,
    date date NOT NULL,
    shift_id smallint NOT NULL,
    accepted_quanity integer NOT NULL,
```

```
    rejected_quantity smallint DEFAULT 0,
    CONSTRAINT produces_pkey PRIMARY KEY (product_id, date, shift_id),
    CONSTRAINT pro FOREIGN KEY (product_id)
        REFERENCES "Products" ("Product_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT shift FOREIGN KEY (shift_id)
        REFERENCES "Shift" ("Shift_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)

CREATE TABLE IF NOT EXISTS "Ad_impact"
(
    "Date" date NOT NULL,
    "Product_id" integer NOT NULL,
    "Prev_Month_Sale" integer DEFAULT 0,
    "Curr_Month_Sale" integer DEFAULT 0,
    CONSTRAINT "Ad_impact_pkey" PRIMARY KEY ("Date", "Product_id"),
    CONSTRAINT pr FOREIGN KEY ("Product_id")
        REFERENCES "Products" ("Product_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE
)

CREATE TABLE IF NOT EXISTS "Orders&Delivery"
(
    "Vechile_number" integer NOT NULL,
    "Buyer_id" integer NOT NULL,
    "Product_id" integer NOT NULL,
    "Date" date NOT NULL,
    "Quantity" integer NOT NULL,
    "Payment_status" character varying COLLATE pg_catalog."default" NOT
NULL,
    CONSTRAINT "Orders&Delivery_pkey" PRIMARY KEY ("Vechile_number",
"Buyer_id", "Product_id", "Date"),
    CONSTRAINT buy FOREIGN KEY ("Buyer_id")
        REFERENCES "Buyers" ("Buyers_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
```

```
    CONSTRAINT prods FOREIGN KEY ("Product_id")
        REFERENCES "Products" ("Product_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT vech FOREIGN KEY ("Vechile_number")
        REFERENCES "Vehicle" ("Vehicle_id") MATCH SIMPLE
        ON UPDATE SET NULL
        ON DELETE SET NULL,
    CONSTRAINT pay_check CHECK ("Payment_status"::text = ANY
(ARRAY['Paid'::character varying, 'Unpaid'::character varying]::text[]))
)

CREATE TABLE IF NOT EXISTS "Mobile_Number"
(
    "Mobile_no" bigint NOT NULL,
    "Employee_id" integer NOT NULL,
    CONSTRAINT "Mobile_Number_pkey" PRIMARY KEY ("Employee_id",
"Mobile_no"),
    CONSTRAINT "Employee" FOREIGN KEY ("Employee_id")
        REFERENCES "Employee" ("Employee_id") MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE CASCADE,
    CONSTRAINT "check" CHECK ("Mobile_no" >= 1000000000 AND "Mobile_no" <=
'9999999999'::bigint)
)

--------------------------------Loading Data---------------------------------

COPY "Raw_materials"("Raw_id","Name","Price")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Raw_materials.csv'
DELIMITER ',' CSV HEADER;

Select * from "Raw_materials";


COPY "Products"("Product_id","Product_name","Product_price")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Product.csv'
DELIMITER ',' CSV HEADER;

Select * from "Products";
```

```
COPY is_used("product_id","Raw_id","Quantity")
FROM 'D:\SEMESTER V\DBMS\Project_Table\is_used.csv'
DELIMITER ',' CSV HEADER;

Select * from is_used;

COPY "Department"("Department_id","Department_Name","Salary","Bonus")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Department.csv'
DELIMITER ',' CSV HEADER;

Select * from "Department";

COPY "Factory_outlet"("Date","Revenue")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Factory_outlet.csv'
DELIMITER ',' CSV HEADER;

Select * from "Factory_outlet";

COPY "Profit/Loss"("Date","Profit","Loss","Total_amount")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Profit_Loss.csv'
DELIMITER ',' CSV HEADER;

Select * from "Profit/Loss";


COPY "Country"("Country_Name","Currency")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Country.csv'
DELIMITER ',' CSV HEADER;

Select * from "Country";

COPY "Shift"("Shift_id","Start_time","End_time")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Shift.csv'
DELIMITER ',' CSV HEADER;

Select * from "Shift";
```

```
COPY
"Employee"("Employee_id","Employee_Name","Shift_id","Department_id","Addre
ss","Country_name","Qualification","DOB","Since")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Employee.csv'
DELIMITER ',' CSV HEADER;

Select * from "Employee";

COPY
"Attendance"("Employee_id","Month","Leaves","Present","Allowed_leaves","To
tal_amount")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Attendance.csv'
DELIMITER ',' CSV HEADER;

Select * from "Attendance";

COPY "Vehicle"("Vehicle_id","Employee_id","Model","Capacity","Status")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Vehicle.csv'
DELIMITER ',' CSV HEADER;

Select * from "Vehicle";

COPY "Insurance"("Month","Employee_id","Amount","Reason")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Insurance.csv'
DELIMITER ',' CSV HEADER;

Select * from "Insurance";

COPY
"Buyers"("Buyers_id","Buyer_name","Store_name","Store_address","Country_na
me","Points")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Buyers.csv'
DELIMITER ',' CSV HEADER;

Select * from "Buyers";

COPY "Cold_warehouse"(date,"Product_id",shift_id,buyer_id,vehicle_number)
FROM 'D:\SEMESTER V\DBMS\Project_Table\Cold_warehouse.csv'
DELIMITER ',' CSV HEADER;
```

```
Select * from "Cold_warehouse";

COPY
"Machine"("Machine_id","Machine_name",department_id,rpm,total_workig_hours
,"Last_serviced")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Machine.csv'
DELIMITER ',' CSV HEADER;

Select * from "Machine";

COPY produced_by(product_id,machine_id,working_time)
FROM 'D:\SEMESTER V\DBMS\Project_Table\Produced_by.csv'
DELIMITER ',' CSV HEADER;

Select * from produced_by;


COPY produces(product_id,date,shift_id,accepted_quanity,rejected_quantity)
FROM 'D:\SEMESTER V\DBMS\Project_Table\Produces.csv'
DELIMITER ',' CSV HEADER;

Select * from produces;

COPY "Ad_impact"("Date","Product_id","Prev_Month_Sale","Curr_Month_Sale")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Ad_impact.csv'
DELIMITER ',' CSV HEADER;

Select * from "Ad_impact";

COPY
"Orders&Delivery"("Vechile_number","Buyer_id","Product_id","Date","Quantit
y","Payment_status")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Orders&Delivery.csv'
DELIMITER ',' CSV HEADER;

Select * from "Orders&Delivery";

COPY "Mobile_Number"("Mobile_no","Employee_id")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Mobile_no.csv'
DELIMITER ',' CSV HEADER;
```

```
Select * from "Mobile_Number";
```

# Snapshots of Relation created along with tuples:-

Table 1: Raw_materials

Query  Query History

```
16  COPY factory_db."Raw_materials"("Raw_id","Name","Price")
17  FROM 'D:\SEMESTER V\DBMS\Project_Table\Raw_materials.csv'
18  DELIMITER ',' CSV HEADER;
19
20  Select * from factory_db."Raw_materials";
```

Data output  Messages  Notifications

| | Raw_id [PK] integer | Name character varying | Price smallint |
|---|---|---|---|
| 1 | 1 | Milk | 50 |
| 2 | 2 | Cream | 55 |
| 3 | 3 | Butter | 72 |
| 4 | 4 | Milk solids | 56 |
| 5 | 5 | Sugar | 85 |
| 6 | 6 | Emulsifier | 56 |
| 7 | 7 | Stabilizer | 70 |
| 8 | 8 | Flavours | 33 |
| 9 | 9 | Colours | 19 |
| 10 | 10 | Fruit | 40 |
| 11 | 11 | Nuts | 13 |
| 12 | 12 | Chocolate | 62 |
| 13 | 13 | Almonds | 29 |
| 14 | 14 | Cashews | 73 |

Table 2: Products

```
23  COPY factory_db."Products"("Product_id","Product_name","Product_price")
24  FROM 'D:\SEMESTER V\DBMS\Project_Table\Product.csv'
25  DELIMITER ',' CSV HEADER;
26
27  Select * from factory_db."Products";
```

Data output | Messages | Notifications

| | Product_id [PK] integer | Product_name character varying | Product_price integer |
|---|---|---|---|
| 1 | 1 | Acai Berry | 86 |
| 2 | 2 | Almond | 78 |
| 3 | 3 | Apple | 14 |
| 4 | 4 | Apricot | 41 |
| 5 | 5 | Asparagus | 89 |
| 6 | 6 | Baci | 78 |
| 7 | 7 | Bacon | 47 |
| 8 | 8 | Balsamic | 58 |
| 9 | 9 | Banana | 75 |
| 10 | 10 | Barbeque | 13 |
| 11 | 11 | Basil | 91 |
| 12 | 12 | Beer | 34 |
| 13 | 13 | Berry | 32 |

Total rows: 50 of 50 | Query complete 00:00:00.045

Total number of tuples = 50

Table3: is_used

```
31  COPY factory_db.is_used("product_id","Raw_id","Quantity")
32  FROM 'D:\SEMESTER V\DBMS\Project_Table\is_used.csv'
33  DELIMITER ',' CSV HEADER;
34
35  Select * from factory_db.is_used;
36
```

Data output | Messages | Notifications

| | Raw_id [PK] integer | product_id [PK] integer | Quantity integer |
|---|---|---|---|
| 1 | 1 | 1 | 50 |
| 2 | 2 | 1 | 92 |
| 3 | 3 | 1 | 43 |
| 4 | 4 | 1 | 54 |
| 5 | 5 | 1 | 51 |
| 6 | 6 | 2 | 35 |
| 7 | 7 | 2 | 84 |
| 8 | 8 | 2 | 58 |
| 9 | 9 | 2 | 20 |
| 10 | 10 | 2 | 35 |
| 11 | 11 | 3 | 20 |
| 12 | 12 | 3 | 74 |
| 13 | 13 | 3 | 18 |

Total rows: 250 of 250 | Query complete 00:00:00.050

Total number of tuples = 250

Table 4: Shift

```
39  COPY factory_db."Shift"("Shift_id","Start_time","End_time")
40  FROM 'D:\SEMESTER V\DBMS\Project_Table\Shift.csv'
41  DELIMITER ',' CSV HEADER;
42
43  Select * from factory_db."Shift";
44
```

Data output | Messages | Notifications

| | Shift_id [PK] smallint | Start_time time with time zone | End_time time with time zone |
|---|---|---|---|
| 1 | 1 | 20:55:00+05:30 | 05:04:00+05:30 |
| 2 | 2 | 22:14:00+05:30 | 06:36:00+05:30 |
| 3 | 3 | 16:06:00+05:30 | 04:35:00+05:30 |
| 4 | 4 | 19:42:00+05:30 | 09:00:00+05:30 |
| 5 | 5 | 20:16:00+05:30 | 21:51:00+05:30 |

Table 5: Department

```
COPY factory_db."Department"("Department_id","Department_Name","Salary","Bonus")
FROM 'D:\SEMESTER V\DBMS\Project_Table\Department.csv'
DELIMITER ',' CSV HEADER;

Select * from factory_db."Department";
```

output Messages Notifications

| Department_id [PK] integer | Department_Name character varying | Salary integer | Bonus integer |
|---|---|---|---|
| 1 | Services | 20000 | 20000 |
| 2 | Product Manage... | 50000 | 50000 |
| 3 | Legal | 30000 | 30000 |
| 4 | Engineering | 30000 | 30000 |
| 5 | Human Resources | 30000 | 30000 |
| 6 | Business Develo... | 50000 | 50000 |
| 7 | Sales | 50000 | 50000 |
| 8 | Marketing | 30000 | 30000 |
| 9 | Accounting | 50000 | 50000 |
| 10 | Training | 30000 | 30000 |

Table 6: Factory_outlet

```
Query  Query History
46  COPY factory_db."Factory_outlet"("Date","Revenue")
47  FROM 'D:\SEMESTER V\DBMS\Project_Table\Factory_outlet.csv'
48  DELIMITER ',' CSV HEADER;
49
50  Select * from factory_db."Factory_outlet";
51
52
53  COPY factory_db."Profit/Loss"("Date","Profit","Loss","Total_amount")
54  FROM 'D:\SEMESTER V\DBMS\Project_Table\Profit_Loss.csv'
```

Data output | Messages | Notifications

| | Date [PK] date | Revenue integer |
|---|---|---|
| 1 | 2022-10-08 | 14267 |
| 2 | 2022-08-01 | 13916 |
| 3 | 2022-10-25 | 13109 |
| 4 | 2022-08-31 | 5909 |
| 5 | 2022-06-04 | 11588 |
| 6 | 2022-01-05 | 13562 |
| 7 | 2021-12-01 | 2267 |
| 8 | 2022-05-17 | 9371 |
| 9 | 2022-04-27 | 3207 |
| 10 | 2022-06-21 | 14456 |
| 11 | 2021-11-20 | 3463 |
| 12 | 2022-03-15 | 2640 |
| 13 | 2022-01-27 | 13874 |

Total rows: 44 of 44  Query complete 00:00:00.088

Table 7: Profit/Loss

```
52
53  COPY factory_db."Profit/Loss"("Date","Profit","Loss","Total_amount")
54  FROM 'D:\SEMESTER V\DBMS\Project_Table\Profit_Loss.csv'
55  DELIMITER ',' CSV HEADER;
56
57  Select * from factory_db."Profit/Loss";
58
59
60  COPY factory_db."Country"("Country_Name","Currency")
```

Data output | Messages | Notifications

| | Date [PK] date | Profit integer | Loss integer | Total_amount integer |
|---|---|---|---|---|
| 1 | 2022-10-08 | 6414 | 1025 | 5389 |
| 2 | 2022-08-01 | 7893 | 1980 | 5913 |
| 3 | 2022-10-25 | 9165 | 4292 | 4873 |
| 4 | 2022-08-31 | 9300 | 63 | 9237 |
| 5 | 2022-06-04 | 3901 | 4932 | -1031 |
| 6 | 2022-01-05 | 14820 | 3542 | 11278 |
| 7 | 2021-12-01 | 5508 | 3033 | 2475 |
| 8 | 2022-05-17 | 9282 | 4806 | 4476 |
| 9 | 2022-04-27 | 1064 | 2759 | -1695 |
| 10 | 2022-06-21 | 1482 | 838 | 644 |
| 11 | 2021-11-20 | 4880 | 2895 | 1985 |
| 12 | 2022-03-15 | 11194 | 4892 | 6302 |
| 13 | 2022-01-27 | 4970 | 298 | 4672 |

Total rows: 44 of 44  Query complete 00:00:00.068

Total number of tuples = 44

Table 8: Country

```
57  Select * from factory_db."Profit/Loss";
58
59
60  COPY factory_db."Country"("Country_Name","Currency")
61  FROM 'D:\SEMESTER V\DBMS\Project_Table\Country.csv'
62  DELIMITER ',' CSV HEADER;
63
64  Select * from factory_db."Country";
65
```

Data output | Messages | Notifications

| | Country_Name [PK] character varying | Currency character varying |
|---|---|---|
| 1 | AFGHANISTAN | Afghani |
| 2 | Ã…LAND ISLANDS | Euro |
| 3 | ALBANIA | Lek |
| 4 | ALGERIA | Algerian Dinar |
| 5 | AMERICAN SAMOA | US Dollar |
| 6 | ANDORRA | Euro |
| 7 | ANGOLA | Kwanza |
| 8 | ANGUILLA | East Caribbean D… |
| 9 | ANTARCTICA | No universal curr… |
| 10 | ANTIGUA AND BARB… | East Caribbean D… |
| 11 | ARGENTINA | Argentine Peso |
| 12 | ARMENIA | Armenian Dram |
| 13 | ARUBA | Aruban Florin |

Total rows: 253 of 253 Query complete 00:00:01.602

Total number of tuples = 253

Table 9: Employee

Query | Query History | Scratch Pad

```
65
66
67
68 COPY factory_db."Employee"("Employee_id","Employee_Name","Shift_id","Department_id","Address","
69 FROM 'D:\SEMESTER V\DBMS\Project_Table\Employee.csv'
70 DELIMITER ',' CSV HEADER;
71
72 Select * from factory_db."Employee";
73
```

Data output | Messages | Notifications

| | Employee_id [PK] integer | Employee_Name character varying | Shift_id integer | Department_id integer | Address character varying | Country_name character varying | Qualification character varying | DOB date | since integer |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Kenny | 2 | 1 | 470 School Park | INDIA | G12 | 1994-05-17 | 2000 |
| 2 | 2 | Mercy | 4 | 8 | 84462 Loftsgord... | INDIA | G10 | 1973-09-20 | 2010 |
| 3 | 3 | Lou | 3 | 2 | 7 Bluestem Way | INDIA | MSc | 1970-11-20 | 1990 |
| 4 | 4 | Rosana | 1 | 9 | 627 Oxford Aven... | INDIA | BSc | 1989-02-27 | 1988 |
| 5 | 5 | Margette | 5 | 4 | 66568 3rd Hill | INDIA | BSc | 1968-06-27 | 2011 |
| 6 | 6 | Dalston | 2 | 1 | 21 Golden Leaf D... | INDIA | G10 | 1976-04-17 | 2006 |
| 7 | 7 | Orlando | 2 | 1 | 3 Lerdahl Circle | INDIA | BSc | 1979-11-08 | 2005 |
| 8 | 8 | Winnie | 3 | 10 | 31 Fairview Aven... | INDIA | MSc | 1979-05-31 | 1997 |
| 9 | 9 | Stacey | 5 | 10 | 21396 Springvie... | INDIA | G12 | 1986-01-30 | 2009 |
| 10 | 10 | Bella | 5 | 7 | 4359 Brentwood ... | INDIA | MSc | 1982-08-07 | 1995 |
| 11 | 11 | Theodor | 2 | 9 | 804 Butterfield C... | INDIA | G12 | 1990-01-19 | 2011 |
| 12 | 12 | Ema | 5 | 5 | 19041 Hayes Pla... | INDIA | G10 | 1969-04-29 | 2003 |
| 13 | 13 | Nicky | 4 | 2 | 520 Hansons Drive | INDIA | G12 | 1963-11-02 | 1993 |

Total rows: 99 of 99 | Query complete 00:00:00.062

Total number of tuples = 99

Table 10: Attendance - Total number of tuples = 100

```
72 Select * from factory_db."Employee";
73
74
75 COPY factory_db."Attendence"("Employee_id","Month","Leaves","Present","Allowed_leaves","Total_a
76 FROM 'D:\SEMESTER V\DBMS\Project_Table\Attendance.csv'
77 DELIMITER ',' CSV HEADER;
78
79 Select * from factory_db."Attendence";
80
```

Data output | Messages | Notifications

| | Employee_id integer | Leaves smallint | Present smallint | Allowed_leaves smallint | Total_amount integer | Month character varying |
|---|---|---|---|---|---|---|
| 1 | 1 | 18 | 12 | 5 | 30000 | January |
| 2 | 1 | 0 | 15 | 5 | 30000 | February |
| 3 | 1 | 13 | 16 | 5 | 30000 | March |
| 4 | 1 | 19 | 19 | 5 | 30000 | April |
| 5 | 1 | 11 | 14 | 5 | 30000 | May |
| 6 | 1 | 20 | 15 | 5 | 30000 | June |
| 7 | 1 | 16 | 15 | 5 | 30000 | July |
| 8 | 1 | 8 | 20 | 5 | 30000 | August |
| 9 | 1 | 10 | 17 | 5 | 30000 | September |
| 10 | 1 | 1 | 19 | 5 | 30000 | October |
| 11 | 1 | 13 | 16 | 5 | 30000 | November |
| 12 | 1 | 13 | 10 | 5 | 30000 | December |
| 13 | 2 | 14 | 20 | 5 | 50000 | January |

Successfully run. Total query runtime:

Total rows: 1000 of 1000 | Query complete 00:00:00.068

## Table 11:Vehicle

```
Query  Query History
79  Select * from factory_db."Attendence";
80
81
82  COPY factory_db."Vehicle"("Vehicle_id","Employee_id","Model","Capacity","Status")
83  FROM 'D:\SEMESTER V\DBMS\Project_Table\Vehicle.csv'
84  DELIMITER ',' CSV HEADER;
85
86  Select * from factory_db."Vehicle";
87
```

Data output | Messages | Notifications

| | Vehicle_id [PK] integer | Employee_id integer | Model character varying | Capacity integer | Status character varying |
|---|---|---|---|---|---|
| 1 | 1 | 63 | 2009 | 5000 | available |
| 2 | 2 | 97 | 2007 | 1000 | available |
| 3 | 3 | 47 | 2006 | 5000 | busy |
| 4 | 4 | 11 | 2007 | 2000 | busy |
| 5 | 5 | 49 | 1994 | 2000 | available |
| 6 | 6 | 73 | 2011 | 2000 | available |
| 7 | 7 | 24 | 1990 | 1000 | available |
| 8 | 8 | 53 | 2000 | 5000 | available |
| 9 | 9 | 2 | 1993 | 2000 | available |
| 10 | 10 | 31 | 2006 | 1000 | busy |
| 11 | 11 | 40 | 2001 | 2000 | busy |
| 12 | 12 | 74 | 1972 | 5000 | busy |
| 13 | 13 | 37 | 2000 | 1000 | busy |

Successfully run. To

Total rows: 50 of 50  Query complete 00:00:00.057

Total number of tuples = 50

## Table 12: Buyers - Total number of tuples = 50

```
Query  Query History
87
88
89  COPY factory_db."Buyers"("Buyers_id","Buyer_name","Store_name","Store_address","Country_name","
90  FROM 'D:\SEMESTER V\DBMS\Project_Table\Buyers.csv'
91  DELIMITER ',' CSV HEADER;
92
93  Select * from factory_db."Buyers";
94
```

Data output | Messages | Notifications

| | Buyers_id [PK] integer | Buyer_name character varying | Store_name character varying | Store_address character varying | Country_name character varying | Points integer |
|---|---|---|---|---|---|---|
| 1 | 1 | Chandal | Emard Inc | 3075 Internation… | VIET NAM | 33 |
| 2 | 2 | Sander | Romaguera and … | 2 Roth Lane | REPUBLIC OF NO… | 3 |
| 3 | 3 | Rozalie | Kuhn-Mayert | 141 Ridgeview Hill | GAMBIA (THE) | 52 |
| 4 | 4 | Julianne | Treutel-Buckridge | 31 Gateway Center | SAINT LUCIA | 62 |
| 5 | 5 | Jeanine | O'Kon-Langosh | 70102 Westerfiel… | PUERTO RICO | 38 |
| 6 | 6 | Patricia | Schuppe Group | 4 Cordelia Crossi… | TURKEY | 11 |
| 7 | 7 | Elijah | Kemmer Inc | 779 Northport Ro… | CAMBODIA | 84 |
| 8 | 8 | Thacher | Kilback LLC | 3286 Fuller Road | MYANMAR | 1 |
| 9 | 9 | Melli | Borer, McClure a… | 3732 Banding Po… | AFGHANISTAN | 85 |
| 10 | 10 | Shaylynn | Kunde Group | 9 Fremont Court | ZAMBIA | 14 |
| 11 | 11 | Alla | Waelchi-Auer | 9591 Gerald Pass | ISRAEL | 8 |
| 12 | 12 | Nanon | Pacocha-Schmeler | 75963 Browning … | UZBEKISTAN | 8 |
| 13 | 13 | Idette | Bernier-Treutel | 2 Sunnyside Point | AFGHANISTAN | |

Successfully run. Total query runtim

Total rows: 50 of 50  Query complete 00:00:00.054

Table 13: Insurance

```
88
89
90  COPY factory_db."Insurance"("Month","Employee_id","Amount","Reason")
91  FROM 'D:\SEMESTER V\DBMS\Project_Table\Insurance.csv'
92  DELIMITER ',' CSV HEADER;
93
94  Select * from factory_db."Insurance";
95
```

Data output | Messages | Notifications

| | Employee_id integer | Amount bigint | Reason character varying | Month character varying |
|---|---|---|---|---|
| 1 | 45 | 10000 | Sick | July |
| 2 | 1 | 20000 | Sick | April |
| 3 | 95 | 90000 | Sick | December |
| 4 | 95 | 60000 | Injury | August |
| 5 | 85 | 40000 | Sick | August |
| 6 | 40 | 80000 | Sick | December |
| 7 | 43 | 80000 | Injury | November |
| 8 | 59 | 100000 | Injury | October |
| 9 | 91 | 40000 | Injury | March |
| 10 | 91 | 70000 | Injury | August |
| 11 | 12 | 100000 | Accident | February |
| 12 | 58 | 30000 | Accident | March |
| 13 | 42 | 90000 | Sick | August |

Total rows: 50 of 50 | Query complete 00:00:00.079

Total number of tuples = 50

Table 14: Cold_warehouse - Total number of tuples = 50

```
97  COPY factory_db."Cold_warehouse"(date,"Product_id",shift_id,buye
98  FROM 'D:\SEMESTER V\DBMS\Project_Table\Cold_warehouse.csv'
99  DELIMITER ',' CSV HEADER;
100
101 Select * from factory_db."Cold_warehouse";
102
103
104
```

Data output | Messages | Notifications

| | date [PK] date | Product_id [PK] integer | shift_id [PK] integer | buyer_id [PK] integer | vehicle_number [PK] integer |
|---|---|---|---|---|---|
| 1 | 2022-04-18 | 1 | 5 | 35 | 28 |
| 2 | 2022-01-02 | 2 | 4 | 40 | 6 |
| 3 | 2021-12-04 | 3 | 3 | 24 | 10 |
| 4 | 2022-07-30 | 4 | 4 | 33 | 5 |
| 5 | 2022-01-28 | 5 | 3 | 7 | 4 |
| 6 | 2021-12-27 | 6 | 3 | 7 | 46 |
| 7 | 2022-02-11 | 7 | 1 | 43 | 1 |
| 8 | 2022-08-16 | 8 | 1 | 37 | 35 |
| 9 | 2022-09-29 | 9 | 1 | 22 | 34 |
| 10 | 2022-03-22 | 10 | 3 | 40 | 27 |
| 11 | 2022-10-09 | 11 | 2 | 23 | 32 |
| 12 | 2021-12-08 | 12 | 1 | 27 | 40 |
| 13 | 2022-04-18 | 13 | 2 | 7 | 35 |

Total rows: 50 of 50 | Query complete 00:00:00.062

## Table 15: Machine

Query Query History

```
103
104
105 COPY factory_db."Machine"("Machine_id","Machine_name",department_id,rpm,total_workig_ho
106 FROM 'D:\SEMESTER V\DBMS\Project_Table\Machine.csv'
107 DELIMITER ',' CSV HEADER;
108
109 Select * from factory_db."Machine";
110
```

Data output Messages Notifications

| | Machine_id [PK] integer | Machine_name character varying | department_id integer | rpm integer | total_workig_hours integer | Last_serviced date |
|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 7 | 20 | 19 | 2022-03-26 |
| 2 | 2 | 2 | 2 | 10 | 22 | 2022-02-07 |
| 3 | 3 | 3 | 7 | 10 | 10 | 2022-07-18 |
| 4 | 4 | 4 | 9 | 10 | 14 | 2022-07-13 |
| 5 | 5 | 5 | 5 | 10 | 6 | 2022-03-18 |
| 6 | 6 | 6 | 8 | 10 | 23 | 2022-01-22 |
| 7 | 7 | 7 | 10 | 12 | 24 | 2022-10-09 |
| 8 | 8 | 8 | 3 | 15 | 17 | 2022-06-22 |
| 9 | 9 | 9 | 8 | 15 | 16 | 2022-08-19 |
| 10 | 10 | 10 | 7 | 15 | 7 | 2022-06-24 |
| 11 | 11 | 11 | 4 | 15 | 16 | 2022-02-05 |
| 12 | 12 | 12 | 8 | 15 | 20 | 2022-01-28 |
| 13 | 13 | 13 | 4 | 15 | 19 | 2022- |

Successfully run. Total q

## Table 16: produced_by

```
111
112 COPY factory_db.produced_by(product_id,machine_id,working_time)
113 FROM 'D:\SEMESTER V\DBMS\Project_Table\Produced_by.csv'
114 DELIMITER ',' CSV HEADER;
115
116 Select * from factory_db.produced_by;
117
```

Data output Messages Notifications

| | product_id [PK] integer | machine_id [PK] integer | working_time smallint |
|---|---|---|---|
| 1 | 1 | 14 | 20 |
| 2 | 2 | 1 | 13 |
| 3 | 3 | 3 | 21 |
| 4 | 4 | 4 | 11 |
| 5 | 5 | 5 | 9 |
| 6 | 6 | 8 | 20 |
| 7 | 7 | 19 | 15 |
| 8 | 8 | 2 | 8 |
| 9 | 9 | 13 | 22 |
| 10 | 10 | 12 | 19 |
| 11 | 11 | 16 | 21 |

## Table 17: produces

```
119
120 COPY factory_db.produces(product_id,date,shift_id,accepted_quanity,rejected_quantity)
121 FROM 'D:\SEMESTER V\DBMS\Project_Table\Produces.csv'
122 DELIMITER ',' CSV HEADER;
123
124 Select * from factory_db.produces;
125
```

Data output | Messages | Notifications

| | product_id [PK] integer | date [PK] date | shift_id [PK] smallint | accepted_quanity integer | rejected_quantity smallint |
|---|---|---|---|---|---|
| 1 | 1 | 2022-06-26 | 2 | 188 | 19 |
| 2 | 2 | 2022-08-12 | 2 | 182 | 24 |
| 3 | 3 | 2022-10-28 | 2 | 165 | 1 |
| 4 | 4 | 2022-05-16 | 4 | 178 | 30 |
| 5 | 5 | 2022-01-04 | 3 | 181 | 16 |
| 6 | 6 | 2022-02-28 | 3 | 165 | 26 |
| 7 | 7 | 2021-11-26 | 3 | 187 | 15 |
| 8 | 8 | 2022-05-13 | 5 | 167 | 43 |
| 9 | 9 | 2022-06-05 | 4 | 165 | 46 |
| 10 | 10 | 2022-06-23 | 4 | 116 | 18 |
| 11 | 11 | 2022-04-06 | 5 | 177 | 48 |

Successfully run. Total que

Total rows: 50 of 50 | Query complete 00:00:00.726

## Table 18: Ad_impact

```
126 COPY factory_db."Ad_impact"("Date","Product_id","Prev_Month_Sale","Curr_Month_Sale")
127 FROM 'D:\SEMESTER V\DBMS\Project_Table\Ad_impact.csv'
128 DELIMITER ',' CSV HEADER;
129
130 Select * from factory_db."Ad_impact";
131
```

Data output | Messages | Notifications

| | Date [PK] date | Product_id [PK] integer | Prev_Month_Sale integer | Curr_Month_Sale integer |
|---|---|---|---|---|
| 1 | 2022-03-07 | 1 | 3474 | 2375 |
| 2 | 2022-03-31 | 2 | 3010 | 4691 |
| 3 | 2022-02-27 | 3 | 3328 | 2805 |
| 4 | 2022-09-13 | 4 | 1874 | 1128 |
| 5 | 2021-12-12 | 5 | 3975 | 4624 |
| 6 | 2022-09-01 | 6 | 4270 | 3666 |
| 7 | 2021-11-15 | 7 | 2203 | 3605 |
| 8 | 2022-04-06 | 8 | 3313 | 1723 |
| 9 | 2021-12-25 | 9 | 1418 | 3091 |
| 10 | 2022-01-21 | 10 | 3973 | 1592 |
| 11 | 2021-12-17 | 11 | 2712 | 3529 |

Successfully run. Tota

Total rows: 50 of 50 | Query complete 00:00:00.056

Total number of tuples = 50

Table 19: Orders&Delivery

```
134 COPY factory_db."Orders&Delivery"("Vechile_number","Buyer_id","Product_id","Date","Quantity","P
135 FROM 'D:\SEMESTER V\DBMS\Project_Table\Orders&Delivery.csv'
136 DELIMITER ',' CSV HEADER;
137
138 Select * from factory_db."Orders&Delivery";
139
```

Data output | Messages | Notifications

| | Vechile_number [PK] integer | Buyer_id [PK] integer | Product_id [PK] integer | Date [PK] date | Quantity integer | Payment_status character varying |
|---|---|---|---|---|---|---|
| 1 | 15 | 10 | 50 | 2022-03-05 | 2274 | Paid |
| 2 | 6 | 11 | 43 | 2022-01-02 | 4584 | Paid |
| 3 | 13 | 6 | 19 | 2022-08-20 | 3125 | Paid |
| 4 | 5 | 31 | 17 | 2022-01-24 | 4165 | Paid |
| 5 | 16 | 5 | 11 | 2021-11-25 | 1982 | Paid |
| 6 | 11 | 14 | 20 | 2021-12-15 | 3255 | Paid |
| 7 | 18 | 5 | 19 | 2022-05-03 | 3486 | Paid |
| 8 | 17 | 34 | 27 | 2022-03-22 | 4204 | Paid |
| 9 | 14 | 27 | 45 | 2022-05-31 | 3681 | Unpaid |
| 10 | 16 | 50 | 13 | 2022-07-14 | 3685 | Unpaid |
| 11 | 15 | 33 | 24 | 2022-05-08 | 4057 | Unpai |

Successfully run. Total query runtime: 2 secs 2

Total rows: 1000 of 1000 | Query complete 00:00:02.291

Total number of tuples = 1000

Table 20: Mobile_no

```
142 COPY factory_db."Mobile_Number"("Mobile_no","Employee_id")
143 FROM 'D:\SEMESTER V\DBMS\Project_Table\Mobile_no.csv'
144 DELIMITER ',' CSV HEADER;
145
146 Select * from factory_db."Mobile_Number";
147
```

Data output | Messages | Notifications

| | Mobile_no [PK] bigint | Employee_id [PK] integer |
|---|---|---|
| 1 | 3939455822 | 27 |
| 2 | 6058727254 | 56 |
| 3 | 1513073518 | 37 |
| 4 | 2719146997 | 5 |
| 5 | 2772207391 | 82 |
| 6 | 4203582387 | 80 |
| 7 | 5349178687 | 55 |
| 8 | 3511980795 | 8 |
| 9 | 4509296823 | 16 |
| 10 | 1269960399 | 100 |
| 11 | 2866411480 | 36 |

Total rows: 175 of 175 | Query complete 00:00:07.021

Total number of tuples = 175

# SQL Queries:-

1. List down the employee's information whose qualification is Msc.

```
151  SET SEARCH_PATH TO factory_db;
152
153  Select * from factory_db."Employee" where "Qualification"='MSc';
154
```

Data output | Messages | Notifications

| | Employee_id [PK] integer | Employee_Name character varying | Shift_id integer | Department_id integer | Address character varying | Country_name character varying | Qualification character varying | DOB date | since integer |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 3 | Lou | 3 | 2 | 7 Bluestem Way | INDIA | MSc | 1970-11-20 | 1990 |
| 2 | 8 | Winnie | 3 | 10 | 31 Fairview Avenue | INDIA | MSc | 1979-05-31 | 1997 |
| 3 | 10 | Bella | 5 | 7 | 4359 Brentwood Circle | INDIA | MSc | 1982-08-07 | 1995 |
| 4 | 16 | Krystle | 1 | 1 | 2661 Melby Alley | INDIA | MSc | 1975-05-24 | 2012 |
| 5 | 19 | Eleonora | 5 | 7 | 24493 Tennyson Parkway | INDIA | MSc | 1974-01-25 | 1967 |
| 6 | 22 | Clareta | 2 | 6 | 7 Lakewood Way | INDIA | MSc | 1974-06-10 | 2011 |
| 7 | 31 | Georgie | 1 | 9 | 0 Moulton Terrace | INDIA | MSc | 1967-08-17 | 1987 |
| 8 | 32 | Obadias | 3 | 10 | 7549 Comanche Pass | INDIA | MSc | 1977-05-08 | 1993 |
| 9 | 34 | Carolynn | 4 | 10 | 04 Homewood Center | INDIA | MSc | 1990-08-09 | 1988 |
| 10 | 42 | Fidelio | 4 | 1 | 4007 Cambridge Lane | INDIA | MSc | 1981-07-14 | 2006 |
| 11 | 46 | Petunia | 4 | 1 | 32967 Moland Plaza | INDIA | MSc | 1978-02-03 | 1996 |
| 12 | 47 | Tedd | 2 | 2 | 4 Anderson Pass | INDIA | MSc | 1980-11-30 | 2001 |
| 13 | 48 | Jasmin | 1 | 1 | 425 Karstens Lane | INDIA | MSc | 1965-12-01 | 2013 |
| 14 | 51 | Murial | 3 | 9 | 9 Superior Trail | INDIA | MSc | 1977-12-08 | 1990 |
| 15 | 53 | Cullan | 5 | 3 | 10 Riverside Pass | INDIA | MSc | 1966-07-02 | 2009 |

Total rows: 29 of 29 | Query complete 00:00:00.105 | Ln 153, Col

2. Show the departments which have salaries greater than 25000.

```
155  Select * from factory_db."Department" where "Salary">=25000;
156
```

Data output | Messages | Notifications

| | Department_id [PK] integer | Department_Name character varying | Salary integer | Bonus integer |
|---|---|---|---|---|
| 1 | 2 | Product Manage... | 50000 | 50000 |
| 2 | 3 | Legal | 30000 | 30000 |
| 3 | 4 | Engineering | 30000 | 30000 |
| 4 | 5 | Human Resources | 30000 | 30000 |
| 5 | 6 | Business Develo... | 50000 | 50000 |
| 6 | 7 | Sales | 50000 | 50000 |
| 7 | 8 | Marketing | 30000 | 30000 |
| 8 | 9 | Accounting | 50000 | 50000 |
| 9 | 10 | Training | 30000 | 30000 |

3. Show the buyer's IDs whose payment status is unpaid.

```
157  Select "Buyer_id" from factory_db."Orders&Delivery" where "Payment_status"='Unpaid';
158
```

Data output | Messages | Notifications

| | Buyer_id integer |
|---|---|
| 1 | 27 |
| 2 | 50 |
| 3 | 33 |
| 4 | 48 |
| 5 | 39 |
| 6 | 49 |
| 7 | 16 |
| 8 | 7 |
| 9 | 14 |
| 10 | 49 |
| 11 | 42 |

Total rows: 164 of 164 | Query complete 00:00:01.351 | Successfully run. Total query run

4. List down the dates when the current month's sale exceeded previous month's sale.

```
159  Select "Date" from factory_db."Ad_impact" where "Curr_Month_Sale" > "Prev_Month_Sale";
160
```

Data output | Messages | Notifications

| | Date date |
|---|---|
| 1 | 2022-03-31 |
| 2 | 2021-12-12 |
| 3 | 2021-11-15 |
| 4 | 2021-12-25 |
| 5 | 2021-12-17 |
| 6 | 2022-08-18 |
| 7 | 2022-06-09 |
| 8 | 2022-10-06 |
| 9 | 2022-10-06 |
| 10 | 2022-07-03 |
| 11 | 2022-09-17 |

Total rows: 20 of 20 | Query complete 00:00:00.074 | Successfully run. Total qu

5. Show the employee's ID who has claimed insurance for being sick.

Query | Query History | Scratch Pad

```
152
153  Select * from factory_db."Employee" where "Qualification"='MSc';
154
155  Select * from factory_db."Department" where "Salary">=25000;
156
157  Select "Buyer_id" from factory_db."Orders&Delivery" where "Payment_status"='Unpaid';
158
159  Select "Date" from factory_db."Ad_impact" where "Curr_Month_Sale" > "Prev_Month_Sale";
160  SELECT "Employee_id" FROM factory_db."Insurance" WHERE "Reason"='Sick';
```

Data output | Messages | Notifications

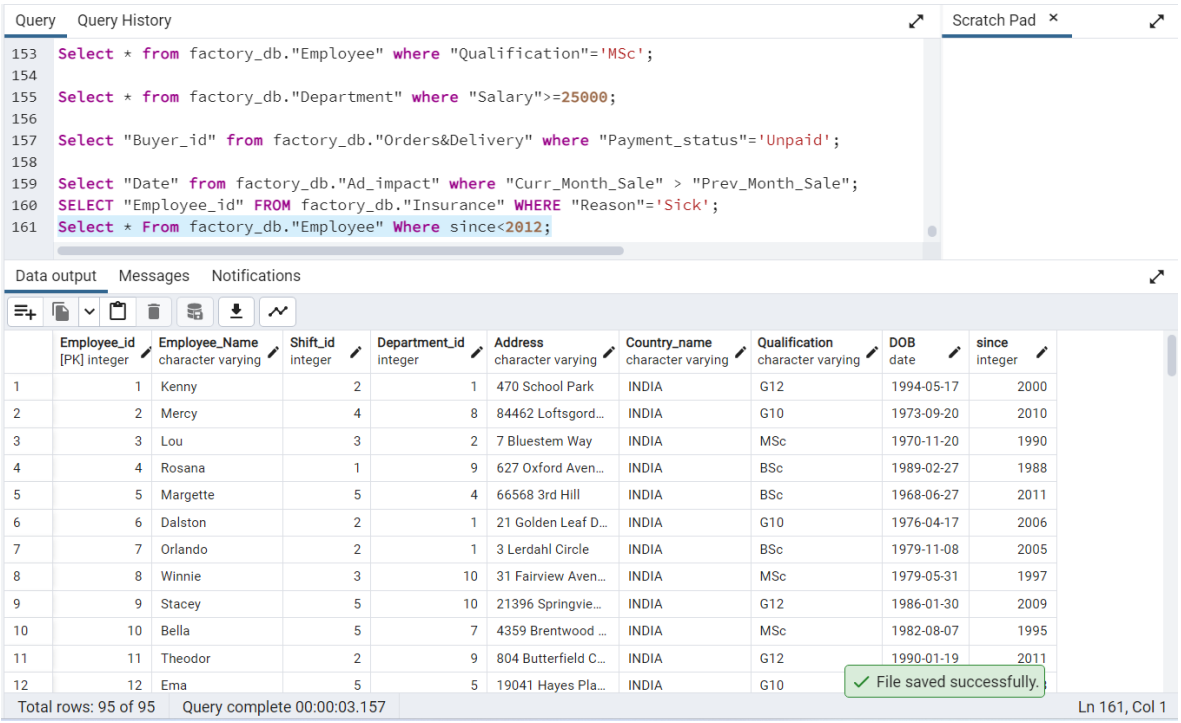
| | Employee_id integer |
|---|---|
| 1 | 45 |
| 2 | 1 |
| 3 | 95 |
| 4 | 85 |
| 5 | 40 |
| 6 | 42 |
| 7 | 27 |
| 8 | 5 |
| 9 | 32 |
| 10 | 17 |
| 11 | 66 |
| 12 | 62 |

Successfully run. Total query runtime: 87 msec. 17 rows affected.

Query returned successfully in 43 msec.

Total rows: 17 of 17 | Query complete 00:00:00.087 | Ln 160, Col 1

6. List down information of Employees working for more than 10 years in the factory.



Query | Query History | Scratch Pad

```
153 Select * from factory_db."Employee" where "Qualification"='MSc';
154
155 Select * from factory_db."Department" where "Salary">=25000;
156
157 Select "Buyer_id" from factory_db."Orders&Delivery" where "Payment_status"='Unpaid';
158
159 Select "Date" from factory_db."Ad_impact" where "Curr_Month_Sale" > "Prev_Month_Sale";
160 SELECT "Employee_id" FROM factory_db."Insurance" WHERE "Reason"='Sick';
161 Select * From factory_db."Employee" Where since<2012;
```

Data output | Messages | Notifications

| | Employee_id [PK] integer | Employee_Name character varying | Shift_id integer | Department_id integer | Address character varying | Country_name character varying | Qualification character varying | DOB date | since integer |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | Kenny | 2 | 1 | 470 School Park | INDIA | G12 | 1994-05-17 | 2000 |
| 2 | 2 | Mercy | 4 | 8 | 84462 Loftsgord... | INDIA | G10 | 1973-09-20 | 2010 |
| 3 | 3 | Lou | 3 | 2 | 7 Bluestem Way | INDIA | MSc | 1970-11-20 | 1990 |
| 4 | 4 | Rosana | 1 | 9 | 627 Oxford Aven... | INDIA | BSc | 1989-02-27 | 1988 |
| 5 | 5 | Margette | 5 | 4 | 66568 3rd Hill | INDIA | BSc | 1968-06-27 | 2011 |
| 6 | 6 | Dalston | 2 | 1 | 21 Golden Leaf D... | INDIA | G10 | 1976-04-17 | 2006 |
| 7 | 7 | Orlando | 2 | 1 | 3 Lerdahl Circle | INDIA | BSc | 1979-11-08 | 2005 |
| 8 | 8 | Winnie | 3 | 10 | 31 Fairview Aven... | INDIA | MSc | 1979-05-31 | 1997 |
| 9 | 9 | Stacey | 5 | 10 | 21396 Springvie... | INDIA | G12 | 1986-01-30 | 2009 |
| 10 | 10 | Bella | 5 | 7 | 4359 Brentwood ... | INDIA | MSc | 1982-08-07 | 1995 |
| 11 | 11 | Theodor | 2 | 9 | 804 Butterfield C... | INDIA | G12 | 1990-01-19 | 2011 |
| 12 | 12 | Ema | 5 | 5 | 19041 Hayes Pla... | INDIA | G10 | | |

File saved successfully.

Total rows: 95 of 95 | Query complete 00:00:03.157 | Ln 161, Col 1

7. List down the flavors of ice cream made by the factory.

```
165  Select "Product_name" From factory_db."Products";
```

Data output | Messages | Notifications

| | Product_name character varying |
|---|---|
| 1 | Acai Berry |
| 2 | Almond |
| 3 | Apple |
| 4 | Apricot |
| 5 | Asparagus |
| 6 | Baci |
| 7 | Bacon |
| 8 | Balsamic |
| 9 | Banana |
| 10 | Barbeque |
| 11 | Basil |
| 12 | Beer |
| 13 | Berry |
| 14 | Bittersweet Choc |

Total rows: 50 of 50 | Query complete 00:00:00.397

8. Show the dates when loss exceeds profit.

```
167  Select "Date" From factory_db."Profit/Loss" WHERE "Loss"> "Profit";
```

Data output | Messages | Notifications

| | Date [PK] date |
|---|---|
| 1 | 2022-06-04 |
| 2 | 2022-04-27 |
| 3 | 2022-08-25 |
| 4 | 2022-09-28 |
| 5 | 2022-03-22 |
| 6 | 2022-02-26 |

9. Show all the available vehicle’s ID.

```
168
169  Select "Vehicle_id" From factory_db."Vehicle" WHERE "Status"='available' ;
```

Data output | Messages | Notifications

| | Vehicle_id [PK] integer |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 3 | 5 |
| 4 | 6 |
| 5 | 7 |
| 6 | 8 |
| 7 | 9 |
| 8 | 14 |
| 9 | 15 |
| 10 | 21 |
| 11 | 22 |
| 12 | 23 |
| 13 | 24 |
| 14 | 25 |

Total rows: 22 of 22 | Query complete 00:00:00.097 | Succes

10.Show the names of the employees who are not from india.

```
171  Select "Employee_id" From factory_db."Employee" WHERE "Country_name" <> 'INDIA' ;
```

Data output | Messages | Notifications

| | Employee_id [PK] integer |
|---|---|
| 1 | 94 |

11.Show all the employee’s salary.

```
173  Select "Employee"."Employee_id","Department"."Salary"
174  From factory_db."Employee" NATURAL JOIN factory_db."Department"
```

Data output | Messages | Notifications

| | Employee_id integer | Salary integer |
|---|---|---|
| 1 | 1 | 20000 |
| 2 | 2 | 30000 |
| 3 | 3 | 50000 |
| 4 | 4 | 50000 |
| 5 | 5 | 30000 |
| 6 | 6 | 20000 |
| 7 | 7 | 20000 |
| 8 | 8 | 30000 |
| 9 | 9 | 30000 |

Total rows: 100 of 100 | Query complete 00:00:01.319 | Succe

12.Show the count of mobile numbers each employee has.

```
176  Select "Mobile_Number"."Employee_id",COUNT("Mobile_no") From factory_db."Mobile_Number"
177  GROUP BY "Mobile_Number"."Employee_id";
```

Data output | Messages | Notifications

| | Employee_id integer | count bigint |
|---|---|---|
| 1 | 74 | 1 |
| 2 | 29 | 2 |
| 3 | 54 | 1 |
| 4 | 71 | 1 |
| 5 | 4 | 1 |
| 6 | 68 | 1 |
| 7 | 34 | 3 |
| 8 | 51 | 2 |
| 9 | 96 | 2 |

Total rows: 81 of 81 | Query complete 00:00:00.650

13.List down all the employees' total leaves in a year.

```
Select "Employee"."Employee_id",SUM("Attendence"."Leaves")
From factory_db."Employee" NATURAL JOIN factory_db."Attendence"
GROUP BY "Employee"."Employee_id";
```

Data output | Messages | Notifications

| | Employee_id [PK] integer | sum bigint |
|---|---|---|
| 1 | 74 | 92 |
| 2 | 54 | 141 |
| 3 | 29 | 121 |
| 4 | 71 | 143 |
| 5 | 68 | 93 |
| 6 | 4 | 130 |
| 7 | 34 | 129 |
| 8 | 51 | 120 |
| 9 | 80 | 99 |

Total rows: 84 of 84 | Query complete 00:00:54.982

14.Show the product's name with its producing machine.

```
Select "Products"."Product_name","produced_by"."machine_id"
From factory_db."Products" NATURAL JOIN factory_db."produced_by";
```

Data output | Messages | Notifications

| | Product_name character varying | machine_id integer |
|---|---|---|
| 1 | Acai Berry | 14 |
| 2 | Almond | 14 |
| 3 | Apple | 14 |
| 4 | Apricot | 14 |
| 5 | Asparagus | 14 |
| 6 | Baci | 14 |
| 7 | Bacon | 14 |
| 8 | Balsamic | 14 |
| 9 | Banana | 14 |

Total rows: 1000 of 1000 | Query complete 00:00:00.094

15.Write the employee's name who has claimed insurance greater than 20000.

```
187  Select "Employee"."Employee_Name"
188  From factory_db."Employee" NATURAL JOIN factory_db."Insurance"
189  WHERE "Insurance"."Amount">20000;
190
```

Data output | Messages | Notifications

| | Employee_Name character varying |
|---|---|
| 1 | Burt |
| 2 | Burt |
| 3 | Kalil |
| 4 | Olympia |
| 5 | Simonne |
| 6 | Lianne |
| 7 | Doretta |
| 8 | Doretta |
| 9 | Ema |

Total rows: 42 of 42 | Query complete 00:00:19.429

16.Show the shift start time and end time with the number of employees working in each shift in ascending order.

```
191  SELECT *
192  FROM factory_db."Shift" as t2 NATURAL JOIN
193  (Select "Employee"."Shift_id" as Shift_id,COUNT("Employee_id") as Employee_count
194  From factory_db."Employee"
195  GROUP BY "Employee"."Shift_id"
196  ORDER BY "Employee"."Shift_id" ASC) as t1
197  WHERE t1.Shift_id=t2."Shift_id";
198
199
200
201
202
203
```

Data output | Messages | Notifications

| | Shift_id smallint | Start_time time with time zone | End_time time with time zone | shift_id integer | employee_count bigint |
|---|---|---|---|---|---|
| 1 | 1 | 20:55:00+05:30 | 05:04:00+05:30 | 1 | 21 |
| 2 | 2 | 22:14:00+05:30 | 06:36:00+05:30 | 2 | 18 |
| 3 | 3 | 16:06:00+05:30 | 04:35:00+05:30 | 3 | 17 |
| 4 | 4 | 19:42:00+05:30 | 09:00:00+05:30 | 4 | 21 |
| 5 | 5 | 20:16:00+05:30 | 21:51:00+05:30 | 5 | 23 |

17. Display the number of employees in the department whose salary is greater than or equal to 30000 in descending order.

```
199 SELECT t2."Department_id",t1.Employee_count
200 FROM factory_db."Department" as t2 NATURAL JOIN
201 (Select "Employee"."Department_id" as Department_id,COUNT("Employee_id") as Employee_count
202 From factory_db."Employee"
203 GROUP BY "Employee"."Department_id"
204 ORDER BY "Employee"."Department_id" ASC) as t1
205 WHERE t1.Department_id=t2."Department_id" AND t2."Salary">=30000
206
207
208
209
```

Data output | Messages | Notifications

| | Department_id [PK] integer | employee_count bigint |
|---|---|---|
| 1 | 2 | 14 |
| 2 | 3 | 11 |
| 3 | 4 | 7 |
| 4 | 5 | 9 |
| 5 | 6 | 8 |
| 6 | 7 | 10 |
| 7 | 8 | 8 |
| 8 | 9 | 9 |
| 9 | 10 | 11 |

Total rows: 9 of 9 | Query complete 00:00:00.066 | Successfully run. Total query

18. Display details of the store whose payment_status is unpaid.

```
208 Select DISTINCT(t1."Store_name", t1."Store_address")
209 from factory_db."Buyers" as t1 natural join factory_db."Orders&Delivery" as t2
210 where t2."Payment_status"='Unpaid';
211
```

Data output | Messages | Notifications

| | row record |
|---|---|
| 1 | ("Bartell LLC","5558 Rutledge Court") |
| 2 | ("Bartell, Haag and Bartell","09 Sutherland Drive") |
| 3 | (Beer-Osinski,"4 Delladonna Crossing") |
| 4 | ("Bergnaum and Sons","7 Spohn Road") |
| 5 | (Bernier-Treutel,"2 Sunnyside Point") |
| 6 | ("Borer, McClure and Lehner","3732 Banding Point") |
| 7 | (Conn-Goyette,"4 Paget Lane") |
| 8 | ("Daniel, Welch and Hyatt","8490 Forster Trail") |
| 9 | (Dare-Huel,"8 Badeau Alley") |

Total rows: 50 of 50 | Query complete 00:00:00.115 | Successfully ru

19. Display name of drivers whose vehicle is in maintenance.

```
229  --19
230  Select DISTINCT(t1."Employee_Name") as driver
231  from factory_db."Employee" as t1 natural join factory_db."Vehicle" as t2
232  where t2."Status"='maintainence';
233
234  --20
```

Data output | Messages | Notifications

| | driver character varying |
|---|---|
| 1 | Pablo |
| 2 | Ulick |

Succ

20. Display the buyer's name with the product's id and the quantity ordered in that year.

```
234  --20
235  SELECT t1."Buyer_name","Orders&Delivery"."Product_id", "Orders&Delivery"."Quantity"
236  FROM (factory_db."Orders&Delivery" NATURAL JOIN factory_db."Buyers" as t1);
237
238
```

Data output | Messages | Notifications

| | Buyer_name character varying | Product_id integer | Quantity integer |
|---|---|---|---|
| 1 | Chandal | 50 | 2274 |
| 2 | Sander | 50 | 2274 |
| 3 | Rozalie | 50 | 2274 |
| 4 | Julianne | 50 | 2274 |
| 5 | Jeanine | 50 | 2274 |
| 6 | Patricia | 50 | 2274 |
| 7 | Elijah | 50 | 2274 |
| 8 | Thacher | 50 | 2274 |
| 9 | Melli | 50 | 2274 |
| 10 | Shaylynn | 50 | 2274 |

Total rows: 1000 of 50000 | Query complete 00:00:00.742

# Functions:-

1. Create a function to know Total profit by adding each day profit with total factory_outlet revenue and subtracting loss of that day.

```
193  CREATE OR REPLACE FUNCTION Total_revenue()
194  RETURNS TABLE(d date, total_revenue int)
195  LANGUAGE 'plpgsql'
196  AS $BODY$
197  BEGIN
198  RETURN QUERY EXECUTE format
199  ('SELECT "Factory_outlet"."Date","Profit/Loss"."Total_amount"+"Factory_outlet"."Revenue"
200   FROM factory_db."Factory_outlet" NATURAL JOIN factory_db."Profit/Loss";');
201  END;
202  $BODY$;
203
204  SELECT factory_db.total_revenue();
205
```

Data output | Messages | Notifications

| | total_revenue record |
|---|---|
| 1 | (2022-10-08,19656) |
| 2 | (2022-08-01,19829) |
| 3 | (2022-10-25,17982) |
| 4 | (2022-08-31,15146) |
| 5 | (2022-06-04,10557) |
| 6 | (2022-01-05,24840) |
| 7 | (2021-12-01,4742) |
| 8 | (2022-05-17,13847) |
| 9 | (2022-04-27,1512) |

Total rows: 44 of 44 | Query complete 00:00:00.054

2. Create a function that calculates the net salary of employees by deducting the salary if leaves are more than total allowed leaves in that month.

```
233 CREATE OR REPLACE FUNCTION net_salary()
234 RETURNS TABLE (Employee_id int, Salary int)
235 LANGUAGE 'plpgsql'
236 AS $BODY$
237 DECLARE R RECORD;
238 BEGIN
239 CREATE TEMP TABLE t1(Employee_id int, salary int) ON COMMIT DROP;
240 FOR R IN (SELECT "Attendence"."Employee_id","Attendence"."Leaves", "Attendence"."Allowed_leaves
241 loop
242 if(R."Leaves">R."Allowed_leaves") then
243 INSERT INTO t1(Employee_id, salary) VALUES (R."Employee_id", R."Total_amount"-((R."Leaves"-R."A
244 else INSERT INTO t1(Employee_id, salary) VALUES (R."Employee_id", R."Total_amount");
245 end if;
246 end loop;
247 RETURN QUERY TABLE t1;
248 END;
249 $BODY$;
250
251 SELECT factory_db.net_salary();
```

Data output | Messages | Notifications

| | net_salary record |
|---|---|
| 14 | (2,25000) |
| 15 | (2,36667) |
| 16 | (2,50000) |

Total rows: 1000 of 1000 | Query complete 00:00:00.110

| | net_salary record |
|---|---|
| 14 | (2,25000) |
| 15 | (2,36667) |
| 16 | (2,50000) |
| 17 | (2,46667) |
| 18 | (2,48334) |
| 19 | (2,50000) |
| 20 | (2,43334) |
| 21 | (2,46667) |
| 22 | (2,50000) |
| 23 | (2,50000) |
| 24 | (2,46667) |
| 25 | (3,31667) |
| 26 | (3,50000) |
| 27 | (3,26667) |
| 28 | (3,33334) |
| 29 | (3,35000) |
| 30 | (3,46667) |
| 31 | (3,40000) |

Total rows: 1000 of 1000 | Query complete

# Trigger Function:-

1. Create a trigger function that increases the points of buyer by 0.5% as it places orders of a particular product.
   *(Points are assigned to each buyer that shows its importance for the factory. The more products buyer buys, more will be their points and hence more valuable for the factory)*

   ☐ Before ordering for products, points of buyer

```
286  SELECT "Buyers"."Buyers_id", "Buyers"."Points" From factory_db."Buyers"
287  WHERE "Buyers"."Buyers_id"=1;
288  insert
```

Data output | Messages | Notifications

| | Buyers_id [PK] integer | Points integer |
|---|---|---|
| 1 | 1 | 33 |

   ☐ Ordering of products

```
SELECT "Buyers"."Buyers_id", "Buyers"."Points" From factory_db."Buyers"
WHERE "Buyers"."Buyers_id"=1;
insert into factory_db."Orders&Delivery" values(1,1,50,'2022-03-05',2000,'Paid');
```

☐ Updated points table

```
286 SELECT "Buyers"."Buyers_id", "Buyers"."Points" From factory_db."Buyers"
287 WHERE "Buyers"."Buyers_id"=1;
288 insert into factory_db."Orders&Delivery" values(1,1,50,'2022-03-05',2000,'Paid');
```

Data output | Messages | Notifications

| | Buyers_id [PK] integer | Points integer |
|---|---|---|
| 1 | 1 | 43 |

As we have placed a trigger on insert in orders&delivery table, the points of buyer increases as he placed the order for any product. [point increases by 0.5% of the quantities of product ordered].

# SECTION 6: Project Code with output screenshots

## “Home Page”


Ayush Gandhi -- Kadam Darji

# Update a new entry in the Product Table:-

Initial Table in Database:

| Product ID | Product Name | Product Price | Action |
|---|---|---|---|
| 1 | Alcai | 900 | Modify Delete |
| 2 | Almond | 60 | Modify Delete |
| 3 | Apple | 20 | Modify Delete |
| 4 | Apricot | 40 | Modify Delete |
| 5 | Asparagus | 90 | Modify Delete |
| 6 | Baci | 78 | Modify Delete |
| 7 | Bacon | 47 | Modify Delete |
| 8 | Balsamic | 58 | Modify Delete |
| 9 | Banana | 75 | Modify Delete |
| 10 | Barbeque | 13 | Modify Delete |
| 11 | Basil | 91 | Modify Delete |
| 12 | Beer | 34 | Modify Delete |
| 13 | Berry | 32 | Modify Delete |
| 14 | Bittersweet Chocolate | 77 | Modify Delete |
| 15 | Black Pepper | 86 | Modify Delete |
| 16 | Blackberry | 6 | Modify Delete |
| 17 | Blue Cheese | 85 | Modify Delete |
| 18 | Blueberry | 50 | Modify Delete |
| 19 | Bourbon | 39 | Modify Delete |
| 20 | Boysenberry | 50 | Modify Delete |
| 21 | Brown Bread | 10 | Modify Delete |
| 22 | Brown Sugar | 77 | Modify Delete |
| 23 | Bumbleberry | 52 | Modify Delete |
| 24 | Butter | 53 | Modify Delete |
| 25 | Buttered Popcorn | 12 | Modify Delete |

Initial Database:

```
1  SELECT * FROM factory_db.products
2  ORDER BY product_id ASC
```

Data output | Messages | Notifications

| | product_id [PK] integer | product_name character varying | product_price integer |
|---|---|---|---|
| 1 | 1 | Alcai | 900 |
| 2 | 2 | Almond | 60 |
| 3 | 3 | Apple | 20 |
| 4 | 4 | Apricot | 40 |
| 5 | 5 | Asparagus | 90 |
| 6 | 6 | Baci | 78 |
| 7 | 7 | Bacon | 47 |
| 8 | 8 | Balsamic | 58 |
| 9 | 9 | Banana | 75 |

Total rows: 50 of 50 Query complete 00:00:00.136

# Updating Alcai Price to 90…


# Updated database:

| Product ID | Product Name | Product Price | Action |
|---|---|---|---|
| 1 | Alcai | 90 | Modify Delete |
| 2 | Almond | 60 | Modify Delete |
| 3 | Apple | 20 | Modify Delete |
| 4 | Apricot | 40 | Modify Delete |
| 5 | Asparagus | 90 | Modify Delete |
| 6 | Baci | 78 | Modify Delete |
| 7 | Bacon | 47 | Modify Delete |

*(in PostgreSQL)*

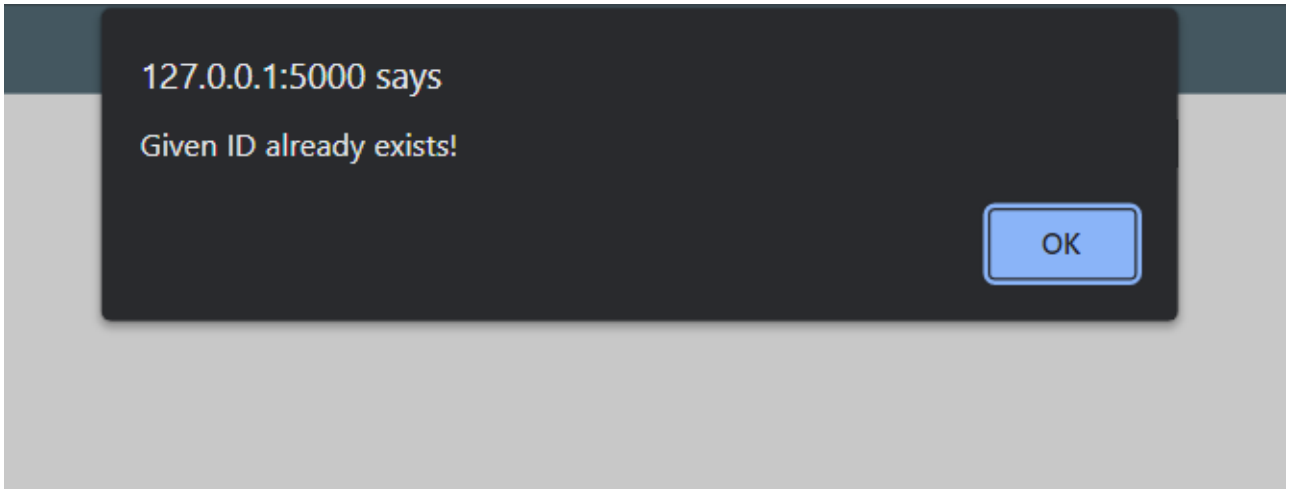
| | product_id [PK] integer | product_name character varying | product_price integer |
|---|---|---|---|
| 1 | 1 | Alcai | 90 |
| 2 | 2 | Almond | 60 |
| 3 | 3 | Apple | 20 |
| 4 | 4 | Apricot | 40 |
| 5 | 5 | Asparagus | 90 |
| 6 | 6 | Baci | 78 |
| 7 | 7 | Bacon | 47 |
| 8 | 8 | Balsamic | 58 |
| 9 | 9 | Banana | 75 |

Total rows: 50 of 50 Query complete 00:00:00.120

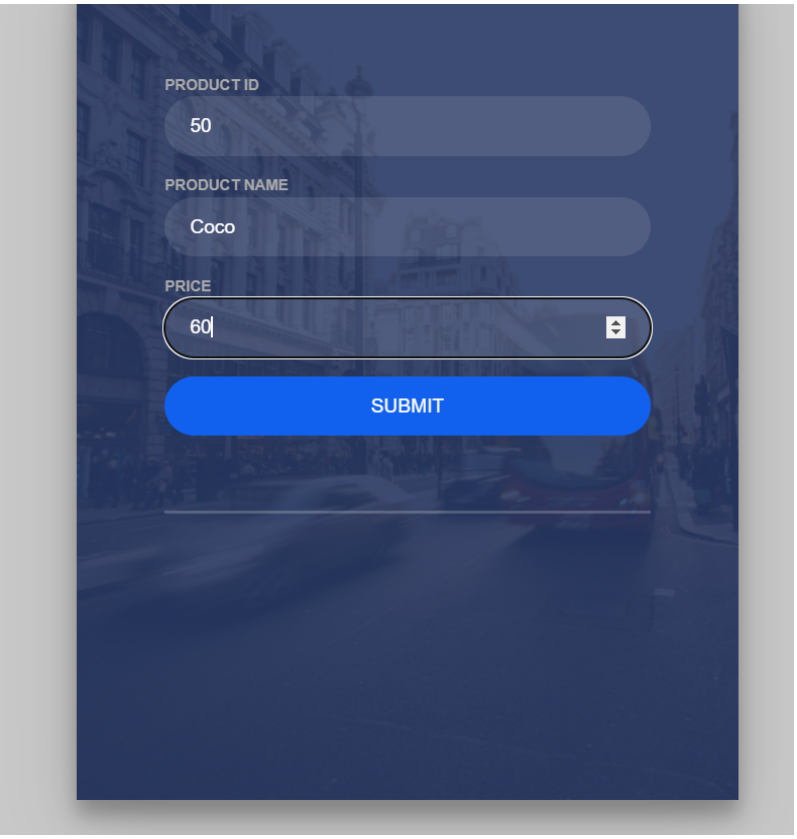
## Insertion:

Inserting already existing raw_id:


The primary key constraint restricts adding new raw material with the same raw_id!



Inserting a new product Coco in the product table:




Successfully Inserted!

| | | | |
|---|---|---|---|
| 29 | Candy Corn | 100 | Modify Delete |
| 30 | Cannoli | 99 | Modify Delete |
| 31 | Cantaloupe | 37 | Modify Delete |
| 32 | Caramel | 80 | Modify Delete |
| 33 | Cardamom | 74 | Modify Delete |
| 34 | Carrot | 70 | Modify Delete |
| 35 | Cheddar | 70 | Modify Delete |
| 36 | Cheese | 47 | Modify Delete |
| 37 | Cherry | 70 | Modify Delete |
| 38 | Cherry Cordial | 82 | Modify Delete |
| 39 | Chilli | 79 | Modify Delete |
| 40 | Chocolate | 71 | Modify Delete |
| 41 | Cinnamon | 34 | Modify Delete |
| 42 | Coca Cola (Coke) | 68 | Modify Delete |
| 43 | Coconut | 91 | Modify Delete |
| 44 | Coffee | 51 | Modify Delete |
| 45 | Cookies | 27 | Modify Delete |
| 46 | Corn | 57 | Modify Delete |
| 47 | Crab | 24 | Modify Delete |
| 48 | Berry | 60 | Modify Delete |
| 49 | Gulab | 40 | Modify Delete |
| 50 | Coco | 60 | Modify Delete |

Add Home

# Deleting Cucumber flavor from the product table:

| | | | |
|---|---|---|---|
| 29 | Candy Corn | 100 | Modify Delete |
| 30 | Cannoli | 99 | Modify Delete |
| 31 | Cantaloupe | 37 | Modify Delete |
| 32 | Caramel | 80 | Modify Delete |
| 33 | Cardamom | 74 | Modify Delete |
| 34 | Carrot | 70 | Modify Delete |
| 35 | Cheddar | 70 | Modify Delete |
| 36 | Cheese | 47 | Modify Delete |
| 37 | Cherry | 70 | Modify Delete |
| 38 | Cherry Cordial | 82 | Modify Delete |
| 39 | Chilli | 79 | Modify Delete |
| 40 | Chocolate | 71 | Modify Delete |
| 41 | Cinnamon | 34 | Modify Delete |
| 42 | Coca Cola (Coke) | 68 | Modify Delete |
| 43 | Coconut | 91 | Modify Delete |
| 44 | Coffee | 51 | Modify Delete |
| 45 | Cookies | 27 | Modify Delete |
| 46 | Corn | 57 | Modify Delete |
| 47 | Crab | 24 | Modify Delete |
| 48 | Berry | 60 | Modify Delete |
| 49 | Gulab | 40 | Modify Delete |
| 50 | Cucumber | 52 | Modify Delete |

Add Home

leteProd/50

# Successfully Deleted!

| | | | |
|---|---|---|---|
| 28 | Cacao Nib | 100 | Modify Delete |
| 29 | Candy Corn | 100 | Modify Delete |
| 30 | Cannoli | 99 | Modify Delete |
| 31 | Cantaloupe | 37 | Modify Delete |
| 32 | Caramel | 80 | Modify Delete |
| 33 | Cardamom | 74 | Modify Delete |
| 34 | Carrot | 70 | Modify Delete |
| 35 | Cheddar | 70 | Modify Delete |
| 36 | Cheese | 47 | Modify Delete |
| 37 | Cherry | 70 | Modify Delete |
| 38 | Cherry Cordial | 82 | Modify Delete |
| 39 | Chilli | 79 | Modify Delete |
| 40 | Chocolate | 71 | Modify Delete |
| 41 | Cinnamon | 34 | Modify Delete |
| 42 | Coca Cola (Coke) | 68 | Modify Delete |
| 43 | Coconut | 91 | Modify Delete |
| 44 | Coffee | 51 | Modify Delete |
| 45 | Cookies | 27 | Modify Delete |
| 46 | Corn | 57 | Modify Delete |
| 47 | Crab | 24 | Modify Delete |
| 48 | Berry | 60 | Modify Delete |
| 49 | Gulab | 40 | Modify Delete |

Add Home

Data output | Messages | Notifications

| product_id [PK] integer | product_name character varying | product_price integer |
|---|---|---|
| 42 | Coca Cola (Coke) | 68 |
| 43 | Coconut | 91 |
| 44 | Coffee | 51 |
| 45 | Cookies | 27 |
| 46 | Corn | 57 |
| 47 | Crab | 24 |
| 48 | Berry | 60 |
| 49 | Gulab | 40 |

Total rows: 49 of 49 Query complete 00:00:00.155

*(in PostgreSQL)*

# Functionality:

## 1. Show the Total employee count in each Shift [Query 16]

| Shift ID | Start Time | End Time | Count |
|---|---|---|---|
| 1 | 20:55:00+05:30 | 05:04:00+05:30 | 21 |
| 2 | 22:14:00+05:30 | 06:36:00+05:30 | 18 |
| 3 | 16:06:00+05:30 | 04:35:00+05:30 | 17 |
| 4 | 19:42:00+05:30 | 09:00:00+05:30 | 21 |
| 5 | 20:16:00+05:30 | 21:51:00+05:30 | 23 |

Home

## 2. Show the total employee count in each department

| Department ID | Count |
|---|---|
| 1 | 13 |
| 2 | 14 |
| 3 | 11 |
| 4 | 7 |
| 5 | 9 |
| 6 | 8 |
| 7 | 10 |
| 8 | 8 |
| 9 | 9 |
| 10 | 11 |

Home

**********Thankyou**********