Anthony Harris

Patrick Healy

Yiduo Wang

Karumbaiah Thimmaiah

# Overview

This Help Desk system is a limited access tool used by administrators to track product cases and employees. Data that is submitted to and approved by system admins is then added to, edited in, or removed from the database.

A small relatively static list of products is purchased from an ever-growing list of customers. These transactions and transactions are stored and indexed in the database. Employee information and changes to it are reported to admins and stored for reference. A subset of employees, salespeople, is separately categorized and referenced. When product issues come up, their descriptions, comments, and resolutions are organized and recorded for future data aggregation.

This function, among many operations, gives value to the system. Browsing cases, customers, and resolutions by various criteria allows relevant information to be referenced so that issues can be solved quickly and accurately. As statuses and circumstances change, so too will this system. When certain events in the database occur, the system will be able to update other parts of the data to logically reflect that change. Furthermore, that logic prevents changes or insertions that do not match the established schema.

That schema allows this Help Desk to safely and consistently house client and company data. That data can be accessed, updated, and deleted concurrently. And from that data, patterns can be recognized. Those patterns can lead to business intelligence that can be used to continually improve this system overall

# Assumptions

1. All database transactions are performed by system administrators.
2. Financial and human resource information is housed in another system.
3. Help Desk is based and exclusively operates in the United States.
4. Orders table only represents sales that have not been returned / refunded.

# Graphical Schema

A close up of a device

Description automatically generated

# Relational Schema

A close up of text on a white background

Description automatically generated

## Entity Sets

### [EMPLOYEES] - <IS A> - [SALESPERSON]

Employees have unique identifiers. Employees also have first name, last name, address, phone, and email attributes that are not necessarily unique but serve to identify the entity. A salesperson is a specialization of employees. Salespersons have their own unique identifier and title. They reference the employee table for personal / general information.

### [CASES] - <HAVE> - [COMMENTS]

Cases have serial ids, summaries, description, and status columns to categorize events, and timestamp columns to store cases chronologically. Cases is a strong entity. Comments (for cases) do not exist without cases, therefore, they are a weak entity. They also have unique identifiers and use text and time as descriptive attributes.

## Relationship Sets

### [CUSTOMERS]- <ORDERS> - [PRODUCTS]

Customers have similar unique and non-unique identifiers as the Employees relation. They also use company and annual income as attributes. Products is a static internal relation that customers choose from. Because many customers can order many products and many products can be ordered by many customers, an Orders junction table is used to track the relationship.

### [CASES] - <HAVE> - [RESOLUTIONS]

Resolutions can exist without cases but they are closely related. A case cannot close without identifying a resolution. They are identified by unique id, name, and steps.

# DDL Statements

drop table Comments;

drop table Cases;

drop table Resolutions;

drop table Salespersons;

drop table Employees;

drop table Orders;

drop table Customers;

drop table Products;

create table Products (

    product\_id serial,

    p\_name varchar (50) not null,

    p\_description text,

    primary key (product\_id)

);

create table Customers (

    customer\_id serial,

    c\_first varchar (20) not null,

    c\_last varchar (20) not null,

    c\_address varchar (100) not null,

    c\_company varchar (50) not null,

    c\_income numeric(10,2) check (c\_income > 0),

    primary key (customer\_id)

);

create table Orders (

    customer\_id int,

    product\_id int,

    o\_date date,

    primary key (customer\_id, product\_id),

    foreign key (customer\_id) references Customers

        on delete cascade,

    foreign key (product\_id) references Products

        on delete cascade

);

create table Employees (

    employee\_id serial,

    e\_first varchar (20) not null,

    e\_last varchar (20) not null,

    e\_address varchar (100) not null,

    e\_phone varchar (20) not null,

    e\_email varchar (50) not null,

    primary key (employee\_id)

);

create table Salespersons (

    salesperson\_id serial,

    s\_title varchar (50),

    employee\_id int,

    primary key (salesperson\_id),

    foreign key (employee\_id) references Employees

        on delete cascade

);

create table Resolutions (

    resolution\_id serial,

    r\_name varchar (50) not null,

    r\_steps text,

    primary key (resolution\_id)

);

create table Cases (

    case\_id serial,

    c\_summary text,

    c\_description text,

    c\_status varchar (20) check (c\_status in ('Open', 'Closed', 'Active')),

    c\_timestart timestamp,

    c\_timeend timestamp,

    employee\_id int,

    product\_id int,

    resolution\_id int,

    primary key (case\_id),

    foreign key (employee\_id) references Employees

        on delete set null,

    foreign key (product\_id) references Products

        on delete set null,

    foreign key (resolution\_id) references Resolutions

        on delete set null

);

create table Comments (

    comment\_id serial,

    c\_text text not null,

    c\_time timestamp,

    case\_id int,

    primary key (comment\_id),

    foreign key (case\_id) references Cases

        on delete cascade

)

## Normalization Analysis

### First Normal Form (1NF)

Each column is meant to accept only one value per record.

### Second Normal Form (2NF)

Database already in first normal form. Each table has a single or composite primary key. They are serialized and cannot be null. Each record is therefore unique. Foreign key constraints maintain referential integrity.

### Third Normal Form (3NF)

Database already in first and second normal form. Comments separate from Cases relation to prevent duplicate information. Addresses attribute in Customer and Employee tables simplified into one column to remove transitive functional dependencies.

Salesperson title split from employees into salesperson table that refers to employee table for personal information. Resolution name and steps split from Cases table into own referential table to remove transitive functional dependencies.

# Front-End Design and Back-End Connection

# System Implementation

# Testing

# Limitations