# Database Design & Develop Document

**for**

# Otaku Haven

**Prepared by Raul Lopez, Gavin Binder, Renato Silva**

**COP3710, Section 1**

# Table of Contents

# 1. Introduction

## 1.1 Executive Summary of Business, Problem, & Solution

Our retail business Otaku Haven specializes in importing asian pop culture goods to the United States. Otaku Haven sells a range of products, including clothing, trading cards, posters, figures, video games, books, albums, and much more. Our business faces many challenges in managing inventory and supplier information. We also face challenges related to managing customer and employee information as well. Due to this, we have created a database to track employees, customers, the invoices created, the items sold, and the suppliers of the items. This database will help increase operational efficiency, help track total stock of inventory, and keep important information to help make our customer experience seamless.

# 2. Design Considerations

## 2.1 Problems and Constraints

One of Otaku Haven's biggest problems is inventory management. Due to having multiple suppliers from different countries, tracking inventory becomes exceptionally difficult. The database aims to make inventory management significantly easier by linking together the products we have with the suppliers and items sold. This will allow for tracking stock, sales history and what items need to be restocked.

Inventory management also leads into supplier management. We have suppliers from many different countries including Japan and Korea and other Asian countries. Keeping track of our suppliers is vital information for contact purposes, we also connect the supplier table to our inventory table, allowing us to keep track of which supplier supplies each item, how long it would take to ship the order, and when it will arrive.

Another problem we face is managing customer and employee information. As we are a retail store, we sometimes need to store customer information such as location and contact for delivery. We also store employee information, that way we can link which employees create an invoice, which day it was generated and its amount, and in case of any issues we will be able to go back to the database and fully understand what was wrong, if it was the address, the employee, the amount paid, etc.

Invoices act as receipts, created by an employee and sent or given to the customer. They help store information to track the entirety of a single purchase. For the customer, this allows them to track their purchase history. This invoices table is linked to the line table, which tracks each individual item that is purchased, rather than a total. Line is connected to the inventory table, and acts as a method of tracking (or updating) the inventory when items are purchased, removed from stock, or new items shipped straight to us from our suppliers.

Another set of problems we face is monetary issues, such as if we can not track how much we are paying for each item, and how much we are selling them for, our investment goes down the drain, we can't be a business and run in the dark, it's practically asking to go bankrupt, and let alone have our employees go wild on us, because if we can not keep track of our money, we most likely will not be able to supply our workers with proper payment and that is its own can of worms that our database allows us to avoid digging deeper into.

## 2.2 Scope and Boundaries

**Inventory**: The database manages the entirety of our product inventory. Each product in inventory should have a unique product id. The inventory table should also store the amount of the item in the inventory, and the price per unit of that product. The inventory table should also store the supplier id to track which suppliers need contact for new inventory. The inventory can store multiple products from a single supplier. The table should be capable of storing the item's name, though it isn't necessary.

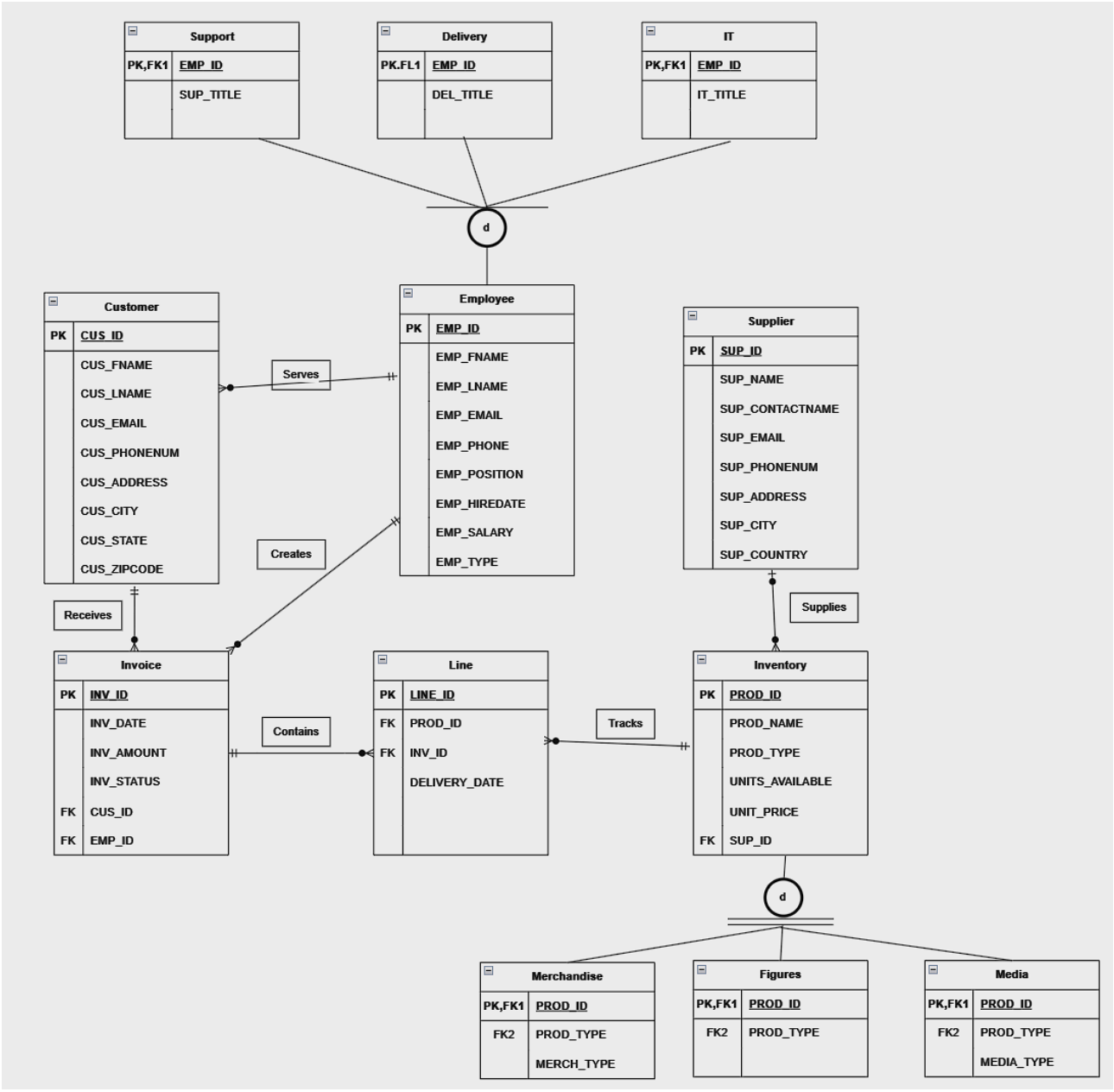
**Customer/Employee Management:** The database will store and manage customer and employee information. There is a mandatory unique customer id for each customer. Customers may have their name, email, and phone number stored. In the event of a delivered purchase, location information may also be stored. Employees also have a mandatory unique id. It is also mandatory that we have their email, phone number, hire date, salary, and position. There are also 3 types of employees, which includes delivery, IT, and Support. Support is customer service, IT is mainly general tech management and website development, delivery employees deal with the delivery of items.

**Sales:** The main table to track sales will be invoice. Each invoice is created by an employee, and received by a customer. Each invoice contains the total purchase amount for all items on the invoice. It may also contain its date and whether the invoice has been paid. Each individual item is tracked in the line table, which tracks the delivery date of the item. The line table is also linked to the inventory through a foreign key, allowing to track the items in inventory that are being shipped, or update the inventory based on the items sold. Each product in inventory is one of three types, merchandise (such as t-shirts, pins, etc), figurines, and media (such as books, dvds, and albums).

**Supplier:** Each supplier will have a mandatory unique supplier id, alongside its contact information. The supplier may also have location information as deemed necessary.

# 3. System Architecture

## 3.1 Enhanced Entity-Relationship Diagrams (EERD)



### 3.2 SQL Dump

Below is the creation of the first three tables of our database:

```
CREATE TABLE `Employee` (
  `EMP_ID` int(11) NOT NULL,
  `EMP_FNAME` varchar(25) NOT NULL,
  `EMP_LNAME` varchar(25) NOT NULL,
  `EMP_EMAIL` varchar(50) NOT NULL,
  `EMP_PHONE` varchar(10) DEFAULT NULL,
  `EMP_POSITION` varchar(30) NOT NULL,
  `EMP_HIREDATE` date NOT NULL,
  `EMP_SALARY` decimal(8,2) NOT NULL,
  `EMP_TYPE` varchar(20) NOT NULL,
  PRIMARY KEY (`EMP_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `Customer` (
  `CUS_ID` int(11) NOT NULL,
  `CUS_FNAME` varchar(25) NOT NULL,
  `CUS_LNAME` varchar(25) NOT NULL,
  `CUS_EMAIL` varchar(50) NOT NULL,
  `CUS_PHONENUM` varchar(10) DEFAULT NULL,
  `CUS_ADDRESS` varchar(30) NOT NULL,
  `CUS_CITY` varchar(20) NOT NULL,
  `CUS_STATE` varchar(20) NOT NULL,
  `CUS_ZIPCODE` int(11) NOT NULL,
  PRIMARY KEY (`CUS_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `Supplier` (
  `SUP_ID` int(11) NOT NULL,
  `SUP_NAME` varchar(50) NOT NULL,
  `SUP_CONTACTNAME` varchar(50) NOT NULL,
  `SUP_EMAIL` varchar(50) NOT NULL,
  `SUP_PHONENUM` varchar(20) NOT NULL,
  `SUP_ADDRESS` varchar(50) DEFAULT NULL,
  `SUP_CITY` varchar(50) DEFAULT NULL,
  `SUP_COUNTRY` varchar(30) NOT NULL,
  PRIMARY KEY (`SUP_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

CREATE TABLE `Invoice` (
  `INV_ID` int(11) NOT NULL,
```

# 4. Queries and Outputs

## 4.1 Filter by Customer

Query to output invoice based on customer name and date purchased:

```
SELECT Invoice.* FROM Invoice
INNER JOIN Customer ON Invoice.CUS_ID = Customer.CUS_ID
WHERE Customer.CUS_FNAME = 'Mylene' AND Customer.CUS_LNAME = 'Crist' AND Invoice.INV_DATE = '2015-09-06';
```

Output | gbinder2005_Final_Project.Invoice

1 row | Tx: Auto | DDL

| | INV_ID | INV_DATE | INV_AMOUNT | INV_STATUS | CUS_ID | EMP_ID |
|---|---|---|---|---|---|---|
| 1 | 0 | 2015-09-06 | 42 | nihil | 2 | 2 |

### 4.2 Filter by Sell Date

Query to show inventory of all products sold between two dates:

```
SELECT Line.* FROM Line
INNER JOIN Invoice ON Line.INV_ID = Invoice.INV_ID
WHERE Invoice.INV_DATE BETWEEN '2010-01-01' AND '2013-01-01';
```

Output | gbinder2005_Final_Project.Line

15 rows | Tx: Auto | DDL

| | LINE_ID | PROD_ID | INV_ID | DELIVERY_DATE |
|---|---|---|---|---|
| 1 | 19 | 98 | 179 | 2008-11-02 |
| 2 | 81 | 69 | 77 | 2009-06-08 |
| 3 | 530 | 26 | 27 | 1987-08-14 |
| 4 | 5957 | 4953139 | 5180963 | 2011-02-05 |
| 5 | 7365 | 41550 | 64582 | 2001-07-12 |
| 6 | 33630 | 966969934 | 927730931 | 2010-04-25 |
| 7 | 69677 | 5171709 | 5719653 | 1971-02-19 |
| 8 | 370552 | 85 | 99 | 2008-08-03 |
| 9 | 531457 | 1601 | 4238 | 2023-07-18 |
| 10 | 1314806 | 4179118 | 3199069 | 2002-02-18 |
| 11 | 37031346 | 458971395 | 801239121 | 2010-05-22 |
| 12 | 59105974 | 78480403 | 224766744 | 2005-02-10 |
| 13 | 71313307 | 87751561 | 251596201 | 2022-02-02 |
| 14 | 78153008 | 480 | 412 | 2022-07-31 |
| 15 | 86371809 | 38 | 48 | 1987-03-16 |

### 4.3 Filter by Product Type

Query to get products of a specific type:

```
SELECT Line.* FROM Line
INNER JOIN Inventory on Line.PROD_ID = Inventory.PROD_ID
WHERE PROD_TYPE='sit';
```

Output | gbinder2005_Final_Project.Line

4 rows | Tx: Auto | DDL

| | LINE_ID | PROD_ID | INV_ID | DELIVERY_DATE |
|---|---|---|---|---|
| 1 | 79 | 43562 | 64765 | 2004-01-13 |
| 2 | 98526 | 51490 | 81297 | 1999-11-11 |
| 3 | 531457 | 1601 | 4238 | 2023-07-18 |
| 4 | 9365625 | 977 | 2725 | 2015-12-11 |

## 4.4 Stored Procedure

Procedure created to get an invoice based on a customer's email:

```
CREATE PROCEDURE INV_BY_EMAIL(
    IN email varchar(50)
)
begin
    SELECT * FROM Invoice
    INNER JOIN Customer on Invoice.CUS_ID=Customer.CUS_ID
    WHERE Customer.CUS_EMAIL=email;
end;

CALL INV_BY_EMAIL( email: 'wvon@example.net');
```

Output | Result 29

1 row

| | INV_ID | INV_DATE | INV_AMOUNT | INV_STATUS | CUS_ID | EMP_ID |
|---|---|---|---|---|---|---|
| 1 | 801239121 | 2012-05-19 | 50360 | nulla | 0 | 0 |

## 4.5 Function to Insert Product

Inserting into inventory a new product, and a query to show it is inserted:

```
INSERT INTO `Inventory` (`PROD_ID`, `PROD_NAME`, `PROD_TYPE`, `UNITS_AVAILABLE`, `UNIT_PRICE`, `SUP_ID`)
VALUES (501, 'AnimeFigure', 'Figure', 1000, '9.99', 5);
SELECT * FROM Inventory WHERE PROD_ID=501;
```

Output | gbinder2005_Final_Project.Inventory

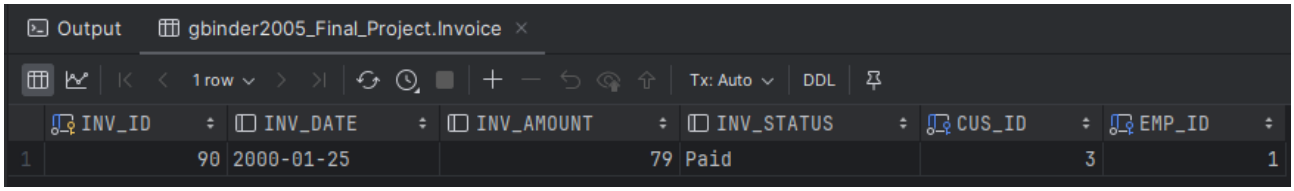
1 row | Tx: Auto | DDL

| | PROD_ID | PROD_NAME | PROD_TYPE | UNITS_AVAILABLE | UNIT_PRICE | SUP_ID |
|---|---|---|---|---|---|---|
| 1 | 501 | AnimeFigure | Figure | 1000 | 9.99 | 5 |

### 4.5 Function to Generate Invoice

Inserting a new invoice, and a query to show it is inserted:

```
INSERT INTO `Invoice` (`INV_ID`, `INV_DATE`, `INV_AMOUNT`, `INV_STATUS`, `CUS_ID`, `EMP_ID`)
VALUES (90, '2000-01-25', 79, 'Paid', 3, 1);
SELECT * FROM Invoice WHERE INV_ID=90;
```

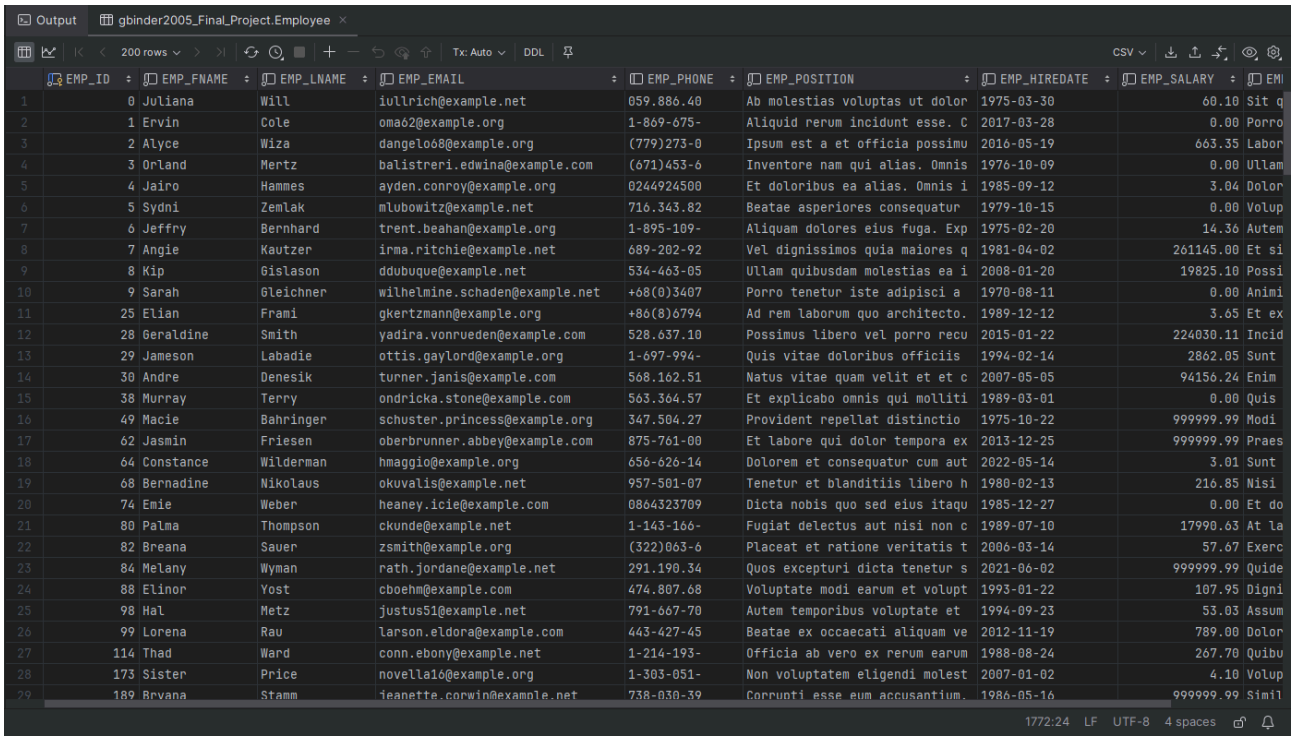



| | INV_ID | INV_DATE | INV_AMOUNT | INV_STATUS | CUS_ID | EMP_ID |
|---|---|---|---|---|---|---|
| 1 | 90 | 2000-01-25 | 79 | Paid | 3 | 1 |

## 4.6 Virtual Table

A full select query showing the Employee table:

```
SELECT * FROM Employee;
```



Output | gbinder2005_Final_Project.Employee

200 rows | Tx: Auto | DDL | CSV

| | EMP_ID | EMP_FNAME | EMP_LNAME | EMP_EMAIL | EMP_PHONE | EMP_POSITION | EMP_HIREDATE | EMP_SALARY | EMI |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | Juliana | Will | iullrich@example.net | 059.886.40 | Ab molestias voluptas ut dolor | 1975-03-30 | 60.10 | Sit q |
| 2 | 1 | Ervin | Cole | oma62@example.org | 1-869-675- | Aliquid rerum incidunt esse. C | 2017-03-28 | 0.00 | Porro |
| 3 | 2 | Alyce | Wiza | dangelo68@example.org | (779)273-0 | Ipsum est a et officia possimu | 2016-05-19 | 663.35 | Labor |
| 4 | 3 | Orland | Mertz | balistreri.edwina@example.com | (671)453-6 | Inventore nam qui alias. Omnis | 1976-10-09 | 0.00 | Ullam |
| 5 | 4 | Jairo | Hammes | ayden.conroy@example.org | 0244924500 | Et doloribus ea alias. Omnis i | 1985-09-12 | 3.04 | Dolor |
| 6 | 5 | Sydni | Zemlak | mlubowitz@example.net | 716.343.82 | Beatae asperiores consequatur | 1979-10-15 | 0.00 | Volup |
| 7 | 6 | Jeffry | Bernhard | trent.beahan@example.org | 1-895-109- | Aliquam dolores eius fuga. Exp | 1975-02-20 | 14.36 | Autem |
| 8 | 7 | Angie | Kautzer | irma.ritchie@example.net | 689-202-92 | Vel dignissimos quia maiores q | 1981-04-02 | 261145.00 | Et si |
| 9 | 8 | Kip | Gislason | ddubuque@example.net | 534-463-05 | Ullam quibusdam molestias ea i | 2008-01-20 | 19825.10 | Possi |
| 10 | 9 | Sarah | Gleichner | wilhelmine.schaden@example.net | +68(0)3407 | Porro tenetur iste adipisci a | 1970-08-11 | 0.00 | Animi |
| 11 | 25 | Elian | Frami | gkertzmann@example.org | +86(8)6794 | Ad rem laborum quo architecto. | 1989-12-12 | 3.65 | Et ex |
| 12 | 28 | Geraldine | Smith | yadira.vonrueden@example.com | 528.637.10 | Possimus libero vel porro recu | 2015-01-22 | 224030.11 | Incid |
| 13 | 29 | Jameson | Labadie | ottis.gaylord@example.org | 1-697-994- | Quis vitae doloribus officiis | 1994-02-14 | 2862.05 | Sunt |
| 14 | 30 | Andre | Denesik | turner.janis@example.com | 568.162.51 | Natus vitae quam velit et et c | 2007-05-05 | 94156.24 | Enim |
| 15 | 38 | Murray | Terry | ondricka.stone@example.com | 563.364.57 | Et explicabo omnis qui molliti | 1989-03-01 | 0.00 | Quis |
| 16 | 49 | Macie | Bahringer | schuster.princess@example.org | 347.504.27 | Provident repellat distinctio | 1975-10-22 | 999999.99 | Modi |
| 17 | 62 | Jasmin | Friesen | oberbrunner.abbey@example.com | 875-761-00 | Et labore qui dolor tempora ex | 2013-12-25 | 999999.99 | Praes |
| 18 | 64 | Constance | Wilderman | hmaggio@example.org | 656-626-14 | Dolorem et consequatur cum aut | 2022-05-14 | 3.01 | Sunt |
| 19 | 68 | Bernadine | Nikolaus | okuvalis@example.net | 957-501-07 | Tenetur et blanditiis libero h | 1980-02-13 | 216.85 | Nisi |
| 20 | 74 | Emie | Weber | heaney.icie@example.com | 0864323709 | Dicta nobis quo sed eius itaqu | 1985-12-27 | 0.00 | Et do |
| 21 | 80 | Palma | Thompson | ckunde@example.net | 1-143-166- | Fugiat delectus aut nisi non c | 1989-07-10 | 17990.63 | At la |
| 22 | 82 | Breana | Sauer | zsmith@example.org | (322)063-6 | Placeat et ratione veritatis t | 2006-03-14 | 57.67 | Exerc |
| 23 | 84 | Melany | Wyman | rath.jordane@example.net | 291.190.34 | Quos excepturi dicta tenetur s | 2021-06-02 | 999999.99 | Quide |
| 24 | 88 | Elinor | Yost | cboehm@example.com | 474.807.68 | Voluptate modi earum et volupt | 1993-01-22 | 107.95 | Digni |
| 25 | 98 | Hal | Metz | justus51@example.net | 791-667-70 | Autem temporibus voluptate et | 1994-09-23 | 53.03 | Assum |
| 26 | 99 | Lorena | Rau | larson.eldora@example.com | 443-427-45 | Beatae ex occaecati aliquam ve | 2012-11-19 | 789.00 | Dolor |
| 27 | 114 | Thad | Ward | conn.ebony@example.net | 1-214-193- | Officia ab vero ex rerum earum | 1988-08-24 | 267.70 | Quibu |
| 28 | 173 | Sister | Price | novella16@example.org | 1-303-051- | Non voluptatem eligendi molest | 2007-01-02 | 4.10 | Volup |
| 29 | 189 | Bryana | Stamm | jeanette.corwin@example.net | 738-030-39 | Corrupti esse eum accusantium. | 1986-05-16 | 999999.99 | Simil |

1772:24 LF UTF-8 4 spaces

## 4.6 Data Dictionary:

Output | information_schema.COLUMNS

53 rows | Tx: Auto | DDL

Reload Page Ctrl+F5

| | table_name | column_name | column_type | is_nullable | column_comment |
|---|---|---|---|---|---|
| 1 | Customer | CUS_ID | ) | NO | |
| 2 | Customer | CUS_FNAME | varchar(25) | NO | |
| 3 | Customer | CUS_LNAME | varchar(25) | NO | |
| 4 | Customer | CUS_EMAIL | varchar(50) | NO | |
| 5 | Customer | CUS_PHONENUM | varchar(10) | YES | |
| 6 | Customer | CUS_ADDRESS | varchar(30) | NO | |
| 7 | Customer | CUS_CITY | varchar(20) | NO | |
| 8 | Customer | CUS_STATE | varchar(20) | NO | |
| 9 | Customer | CUS_ZIPCODE | int(11) | NO | |
| 10 | Delivery | EMP_ID | int(11) | NO | |
| 11 | Delivery | DEL_TITLE | varchar(30) | NO | |
| 12 | Employee | EMP_ID | int(11) | NO | |
| 13 | Employee | EMP_FNAME | varchar(25) | NO | |
| 14 | Employee | EMP_LNAME | varchar(25) | NO | |
| 15 | Employee | EMP_EMAIL | varchar(50) | NO | |
| 16 | Employee | EMP_PHONE | varchar(10) | YES | |
| 17 | Employee | EMP_POSITION | varchar(30) | NO | |
| 18 | Employee | EMP_HIREDATE | date | NO | |
| 19 | Employee | EMP_SALARY | decimal(8,2) | NO | |
| 20 | Employee | EMP_TYPE | varchar(20) | NO | |
| 21 | Figures | PROD_ID | int(11) | NO | |
| 22 | Inventory | PROD_ID | int(11) | NO | |
| 23 | Inventory | PROD_NAME | varchar(100) | YES | |
| 24 | Inventory | PROD_TYPE | varchar(25) | YES | |
| 25 | Inventory | UNITS_AVAILABLE | int(11) | NO | |
| 26 | Inventory | UNIT_PRICE | decimal(5,2) | NO | |
| 27 | Inventory | SUP_ID | int(11) | NO | |
| 28 | Invoice | INV_ID | int(11) | NO | |
| 29 | Invoice | INV_DATE | date | YES | |
| 30 | Invoice | INV_AMOUNT | int(11) | YES | |
| 31 | Invoice | INV_STATUS | varchar(20) | YES | |
| 32 | Invoice | CUS_ID | int(11) | NO | |
| 33 | Invoice | EMP_ID | int(11) | NO | |
| 34 | IT | EMP_ID | int(11) | NO | |
| 35 | IT | IT_TITLE | varchar(30) | NO | |
| 36 | Line | LINE_ID | int(11) | NO | |
| 37 | Line | PROD_ID | int(11) | NO | |
| 38 | Line | INV_ID | int(11) | NO | |
| 39 | Line | DELIVERY_DATE | date | NO | |
| 40 | Media | PROD_ID | int(11) | NO | |
| 41 | Media | MEDIA_TYPE | varchar(25) | YES | |
| 42 | Merchandise | PROD_ID | int(11) | NO | |
| 43 | Merchandise | MERCH_TYPE | varchar(25) | YES | |
| 44 | Supplier | SUP_ID | int(11) | NO | |
| 45 | Supplier | SUP_NAME | varchar(50) | NO | |
| 46 | Supplier | SUP_CONTACTNAME | varchar(50) | NO | |
| 47 | Supplier | SUP_EMAIL | varchar(50) | NO | |
| 48 | Supplier | SUP_PHONENUM | varchar(20) | NO | |
| 49 | Supplier | SUP_ADDRESS | varchar(50) | YES | |
| 50 | Supplier | SUP_CITY | varchar(50) | YES | |
| 51 | Supplier | SUP_COUNTRY | varchar(30) | NO | |
| 52 | Support | EMP_ID | int(11) | NO | |
| 53 | Support | SUP_TITLE | varchar(30) | NO | |

# 5. Conclusion

## 5.1 Closing Remarks

In this business proposal, we have created a model database to solve business problems such as managing inventory, sales, suppliers, employee information, and customer information. The employee table is created to track positions of employees, alongside their salaries, and who creates a specific invoice. Customer information is stored for the ability to contact the customer for issues, alongside location information for delivery. We then use the invoice as a receipt sent to the customer.

Each invoice is primarily used to track the total sales of a customer and whether that customer had paid the full amount or not. The invoice table is expanded in the line table, which specifies which single items were sold, not a total of items. This invoice table then connects to our inventory, allowing us to track which items from inventory are sold, and possibly how frequently a specific item might be sold as well.

Lastly, we have our supplier table. This table is vitally important, as keeping track of dozens of suppliers from different countries is very difficult. This table is connected to our inventory, so we can always know when we get low on items. In conclusion, we have created a database that tracks vital information related to sales and suppliers for retail purposes.

## 5.2 Closing Remarks

Teammate Contribution:

| Gavin Binder | Renato Silva | Raul Lopez |
|---|---|---|
| 35% | 30% | 35% |