# Commissions Database

CMPT 308: Design Project

Erina Caferra

4-24-2016

## Table of Contents

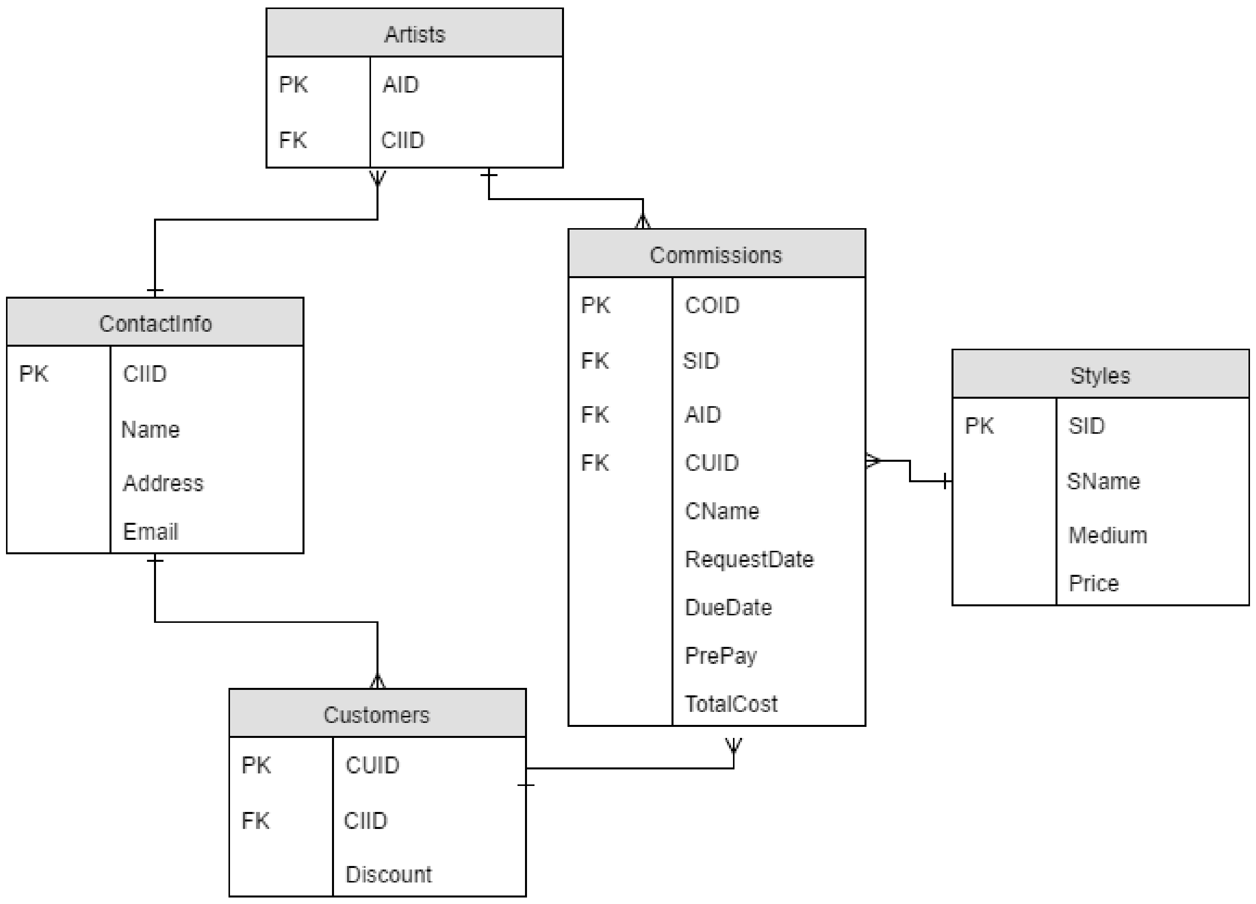
## Overview

This is a database for artistic commissions. It can be difficult to manage commissions for artists and customers, but with a dynamic database such as this one, it will be easy to. This will be a structure that will help organize current and future commissions for artists with similar prices.

## Objectives

The purpose of this document is to outline a database system to record artists' information, customers' information, styles of commissions, and the commissions themselves. This allows customers to be view commission styles and prices offered by artists. Additionally, it will allow for artists to see the commissions that they are currently working on or have to work on for an upcoming due date. This document will provide an overview of the database, along with technical and implementation details including but not limited to: tables and their functional dependencies, views, reports, stored procedures, triggers, and security.

## Entity Relationship Diagram



## Tables

### Contact Info

**Purpose:**

This table stores the information for each person in the commission process. The name, address, and email of each person is recorded here.

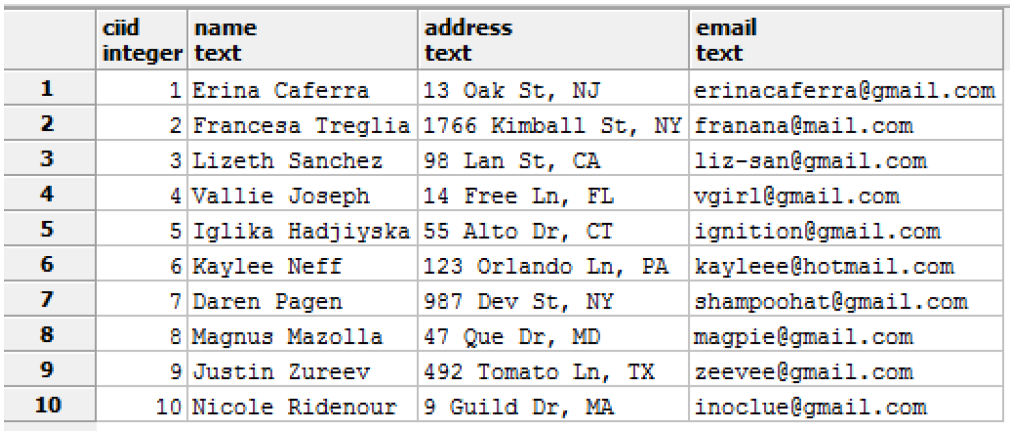
**Create Statement:**

```
CREATE TABLE IF NOT EXISTS ContactInfo(
     ciid      INT NOT NULL,
     name      TEXT NOT NULL,
     address   TEXT,
     email     TEXT NOT NULL,
     PRIMARY KEY(ciid)
);
```

**Functional Dependencies:**

CIID → name, address, email

**Sample Data:**



| | ciid integer | name text | address text | email text |
|---|---|---|---|---|
| 1 | 1 | Erina Caferra | 13 Oak St, NJ | erinacaferra@gmail.com |
| 2 | 2 | Francesa Treglia | 1766 Kimball St, NY | franana@mail.com |
| 3 | 3 | Lizeth Sanchez | 98 Lan St, CA | liz-san@gmail.com |
| 4 | 4 | Vallie Joseph | 14 Free Ln, FL | vgirl@gmail.com |
| 5 | 5 | Iglika Hadjiyska | 55 Alto Dr, CT | ignition@gmail.com |
| 6 | 6 | Kaylee Neff | 123 Orlando Ln, PA | kayleee@hotmail.com |
| 7 | 7 | Daren Pagen | 987 Dev St, NY | shampoohat@gmail.com |
| 8 | 8 | Magnus Mazolla | 47 Que Dr, MD | magpie@gmail.com |
| 9 | 9 | Justin Zureev | 492 Tomato Ln, TX | zeevee@gmail.com |
| 10 | 10 | Nicole Ridenour | 9 Guild Dr, MA | inoclue@gmail.com |

**Artists**

**Purpose:**

This table stores information for each artist. There is only information to tell which people are artists, having CIID and AID.

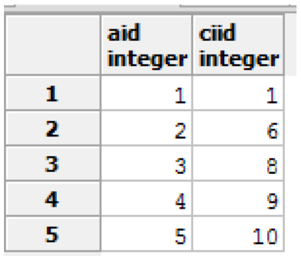
**Create Statement:**

```
CREATE TABLE IF NOT EXISTS Artists(
     aid        INT NOT NULL,
     ciid       INT NOT NULL REFERENCES ContactInfo(CIID),
     PRIMARY KEY(aid)
);
```

**Functional Dependencies:**

AID → CIID

**Sample Data:**



| | aid integer | ciid integer |
|---|---|---|
| 1 | 1 | 1 |
| 2 | 2 | 6 |
| 3 | 3 | 8 |
| 4 | 4 | 9 |
| 5 | 5 | 10 |

## Customers

**Purpose:**

This table provides information about customers. There is information to tell which people are customers and (if they have one) the amount of discount they have.

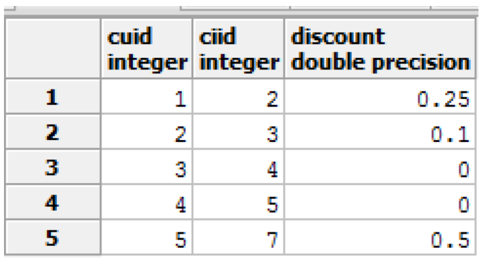
**Create Statement:**

```
CREATE TABLE IF NO EXISTS Customers(
     cuid      INT NOT NULL,
     ciid      INT NOT NULL REFERENCES ContactInfo(CIID),
     discount  FLOAT,
     PRIMARY KEY(cuid)
);
```

**Functional Dependencies:**

CUID → CIID, Discount

**Sample Data:**



|   | cuid integer | ciid integer | discount double precision |
|---|---|---|---|
| 1 | 1 | 2 | 0.25 |
| 2 | 2 | 3 | 0.1 |
| 3 | 3 | 4 | 0 |
| 4 | 4 | 5 | 0 |
| 5 | 5 | 7 | 0.5 |

## <u>Styles</u>

**Purpose:**

This table provides information about styles available for commissions. It contains the name of the style, the medium of the style, and the base price that the style starts at.

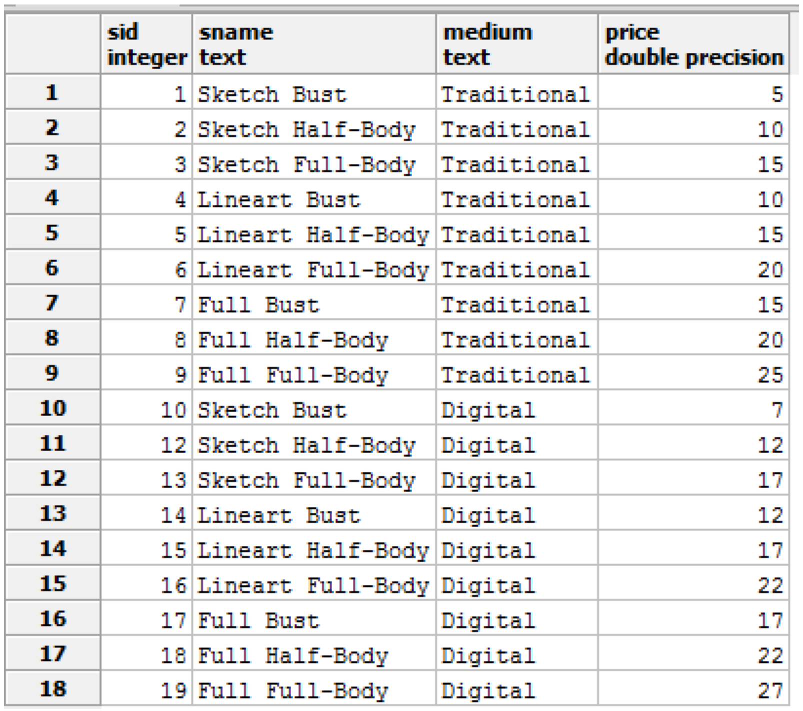
**Create Statement:**

```
CREATE TABLE IF NOT EXISTS Styles(
     sid  INT NOT NULL,
     sname     TEXT NOT NULL,
     medium    TEXT NOT NULL,
     price     DOUBLE NOT NULL,
     PRIMARY KEY(SID)
);
```

**Functional Dependencies:**

CUID → CIID

**Sample Data:**



| | sid integer | sname text | medium text | price double precision |
|---|---|---|---|---|
| 1 | 1 | Sketch Bust | Traditional | 5 |
| 2 | 2 | Sketch Half-Body | Traditional | 10 |
| 3 | 3 | Sketch Full-Body | Traditional | 15 |
| 4 | 4 | Lineart Bust | Traditional | 10 |
| 5 | 5 | Lineart Half-Body | Traditional | 15 |
| 6 | 6 | Lineart Full-Body | Traditional | 20 |
| 7 | 7 | Full Bust | Traditional | 15 |
| 8 | 8 | Full Half-Body | Traditional | 20 |
| 9 | 9 | Full Full-Body | Traditional | 25 |
| 10 | 10 | Sketch Bust | Digital | 7 |
| 11 | 12 | Sketch Half-Body | Digital | 12 |
| 12 | 13 | Sketch Full-Body | Digital | 17 |
| 13 | 14 | Lineart Bust | Digital | 12 |
| 14 | 15 | Lineart Half-Body | Digital | 17 |
| 15 | 16 | Lineart Full-Body | Digital | 22 |
| 16 | 17 | Full Bust | Digital | 17 |
| 17 | 18 | Full Half-Body | Digital | 22 |
| 18 | 19 | Full Full-Body | Digital | 27 |

## Commissions

**Purpose:**

This table provides information about commissions. It contains the style ID, Artist ID, Customer ID, Commission Name, Request Date, an optional Due Date, down payment PrePay, and the Total Cost of the entire commission.

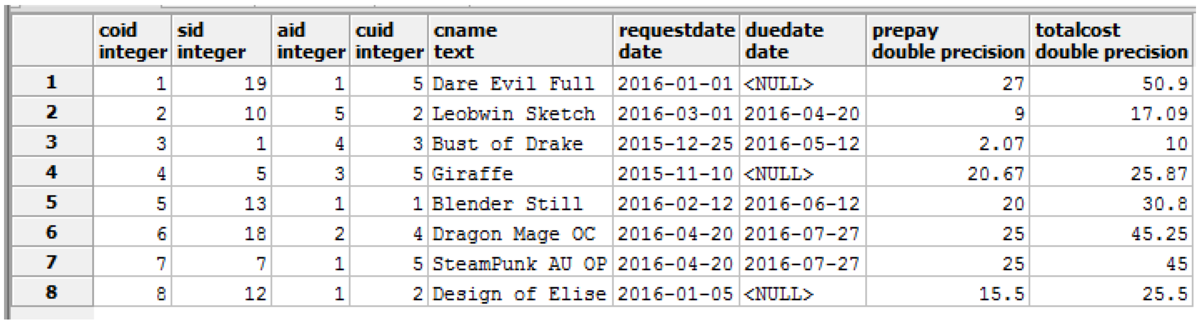
**Create Statement:**

```
CREATE TABLE IF NOT EXISTS Commissions(
     coid            INT NOT NULL,
     sid             INT NOT NULL REFERENCES Styles(sid),
     aid             INT NOT NULL REFERENCES Artists(aid),
     cuid            INT NOT NULL REFERENCES Customers(cuid),
     cname           TEXT NOT NULL,
     requestdate     DATE NOT NULL,
     duedate         DATE,
     prepay          FLOAT NOT NULL,
     totalcost       FLOAT NOT NULL,
     PRIMARY KEY(coid)
);
```

**Functional Dependencies:**

COID → SID, AID, CUID, CName, RequestDate, DueDate, PrePay, TotalCost

**Sample Data:**



| | coid integer | sid integer | aid integer | cuid integer | cname text | requestdate date | duedate date | prepay double precision | totalcost double precision |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 19 | 1 | 5 | Dare Evil Full | 2016-01-01 | <NULL> | 27 | 50.9 |
| 2 | 2 | 10 | 5 | 2 | Leobwin Sketch | 2016-03-01 | 2016-04-20 | 9 | 17.09 |
| 3 | 3 | 1 | 4 | 3 | Bust of Drake | 2015-12-25 | 2016-05-12 | 2.07 | 10 |
| 4 | 4 | 5 | 3 | 5 | Giraffe | 2015-11-10 | <NULL> | 20.67 | 25.87 |
| 5 | 5 | 13 | 1 | 1 | Blender Still | 2016-02-12 | 2016-06-12 | 20 | 30.8 |
| 6 | 6 | 18 | 2 | 4 | Dragon Mage OC | 2016-04-20 | 2016-07-27 | 25 | 45.25 |
| 7 | 7 | 7 | 1 | 5 | SteamPunk AU OP | 2016-04-20 | 2016-07-27 | 25 | 45 |
| 8 | 8 | 12 | 1 | 2 | Design of Elise | 2016-01-05 | <NULL> | 15.5 | 25.5 |

## Views

**commissionsDue**

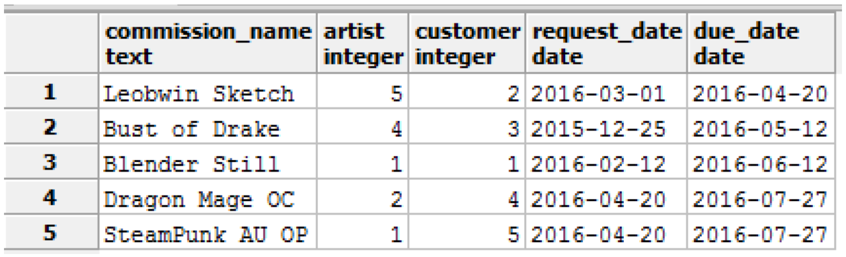
**Purpose:**

This view is created to see all of the commissions that have a due date. It includes the commission name, artist ID, customer ID, request date and date due.

**Create Statement:**

```
CREATE VIEW commissionsDue AS
     SELECT    co.cname AS Commission_Name,
               a.aid AS Artist_ID,
               cu.cuid AS Customer_ID,
               co.requestdate AS Request_Date,
               co.duedate AS Due_Date
     FROM      Commissions co,
               Artists a,
               Customers cu
     WHERE     co.duedate IS NOT NULL
       AND     a.aid = co.aid
       AND     cu.cuid = co.cuid
     ORDER BY  co.duedate ASC;
```

**Sample Data:**



| | commission_name text | artist integer | customer integer | request_date date | due_date date |
|---|---|---|---|---|---|
| 1 | Leobwin Sketch | 5 | 2 | 2016-03-01 | 2016-04-20 |
| 2 | Bust of Drake | 4 | 3 | 2015-12-25 | 2016-05-12 |
| 3 | Blender Still | 1 | 1 | 2016-02-12 | 2016-06-12 |
| 4 | Dragon Mage OC | 2 | 4 | 2016-04-20 | 2016-07-27 |
| 5 | SteamPunk AU OP | 1 | 5 | 2016-04-20 | 2016-07-27 |

## discountPrice

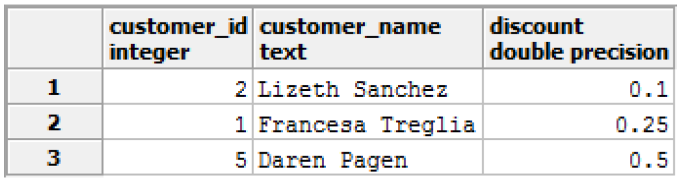
**Purpose:**

This view is created to see the customers who have discounts and what their discounts are. It includes the customer ID, customer name and discount.

**Create Statement:**

```
CREATE VIEW discountPrice AS
     SELECT    cu.cuid AS Customer_ID,
               ci.name AS Customer_Name,
               cu.discount AS Discount
     FROM      Customers cu,
               ContactInfo ci
     WHERE     cu.ciid = ci.ciid
       AND     cu.discount != 0
     ORDER BY  cu.discount ASC;
```

**Sample Data:**



| | customer_id integer | customer_name text | discount double precision |
|---|---|---|---|
| 1 | 2 | Lizeth Sanchez | 0.1 |
| 2 | 1 | Francesa Treglia | 0.25 |
| 3 | 5 | Daren Pagen | 0.5 |

## finalPrices

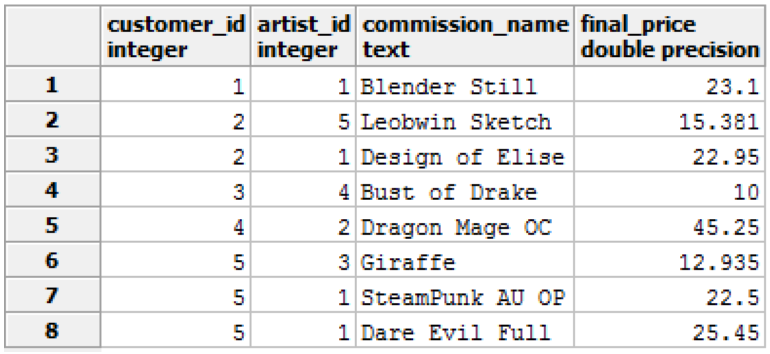
**Purpose:**

This view contains the calculated prices for each commission if the customer has a discount.

**Create Statement:**

```
CREATE OR REPLACE VIEW finalPrices AS
     SELECT    co.cuid AS Customer_ID,
               co.aid AS Artist_ID,
               co.cname AS Commission_Name,
               (co.totalcost-(co.totalcost*cu.discount)) AS
          Final_Price
     FROM      Commissions co
     LEFT JOIN Customers cu ON co.cuid= cu.cuid
     ORDER BY co.cuid ASC;
```

**Sample Data:**



| | customer_id | artist_id | commission_name | final_price |
|---|---|---|---|---|
| | integer | integer | text | double precision |
| 1 | 1 | 1 | Blender Still | 23.1 |
| 2 | 2 | 5 | Leobwin Sketch | 15.381 |
| 3 | 2 | 1 | Design of Elise | 22.95 |
| 4 | 3 | 4 | Bust of Drake | 10 |
| 5 | 4 | 2 | Dragon Mage OC | 45.25 |
| 6 | 5 | 3 | Giraffe | 12.935 |
| 7 | 5 | 1 | SteamPunk AU OP | 22.5 |
| 8 | 5 | 1 | Dare Evil Full | 25.45 |

## Reports

**<u>Total Income for Each Artist (with discounts)</u>**

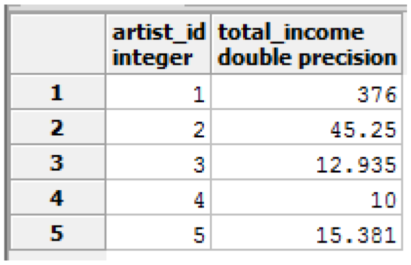
**Purpose:**

This is to show the total income for each artist. This includes the discounts that each customer has.

**Query:**

```
SELECT     co.aid AS Artist_ID,
           SUM(fp.Final_Price) AS Total_Income
FROM       Commissions co
LEFT JOIN finalPrices fp ON co.aid = fp.Artist_ID
GROUP BY co.aid
ORDER BY co.aid ASC;
```

**Sample:**



| | artist_id integer | total_income double precision |
|---|---|---|
| 1 | 1 | 376 |
| 2 | 2 | 45.25 |
| 3 | 3 | 12.935 |
| 4 | 4 | 10 |
| 5 | 5 | 15.381 |

## <u>Total Income for Each Artist (without discounts)</u>

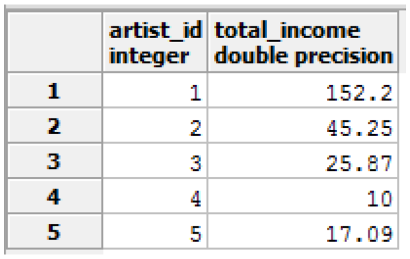
**Purpose:**

This is to show the total income for each artist. This does not include the discounts that each customer has.

**Query:**

```
SELECT    co.aid AS Artist_ID,
          SUM(co.totalcost) AS Total_Income
FROM      Commissions co
LEFT JOIN Artists a ON co.aid = a.aid
GROUP BY co.aid
ORDER BY co.aid ASC;
```

**Sample:**



|   | artist_id integer | total_income double precision |
|---|---|---|
| 1 | 1 | 152.2 |
| 2 | 2 | 45.25 |
| 3 | 3 | 25.87 |
| 4 | 4 | 10 |
| 5 | 5 | 17.09 |

## Stored Procedures

### ChangeName

**Purpose:**

This is used if a customer or artist needs to change their name in the database.

**Query:**

```
CREATE OR REPLACE FUNCTION changeName(text, text, REFCURSOR)
RETURNS refcursor AS
$$
     DECLARE
          old text := $1;
          new text := $2;
          resultset REFCURSOR := $3;
     BEGIN
          open resultset for
               UPDATE ContactInfo
               SET Name=new
               WHERE Name=old;
          Return resultset;
     END;
$$ LANGUAGE plpgsql;
```

## Triggers

**addCommissions**

**Purpose:**

A trigger for when the name of a person is changed.

**Query:**

```
CREATE TRIGGER addCommissions
AFTER UPDATE ON Commissions
 FOR EACH ROW EXECUTE PROCEDURE changeName();
```

## Security

**Customers**

```
GRANT DELETE ON Commissions TO customers;
GRANT INSERT, SELECT, UPDATE, DELETE ON ContactInfo TO artists;
```

**Artists**

```
GRANT INSERT, SELECT, UPDATE, DELETE ON Commissions TO artists;
GRANT INSERT, SELECT, UPDATE, DELETE ON Artists TO artists;
GRANT INSERT, SELECT, UPDATE, DELETE ON ContactInfo TO artists;
GRANT INSERT, SELECT, UPDATE, DELETE ON Styles TO artists;
```

**Database Administrator Role**

```
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO admin;
```

## Implementation Notes

These are suggestions for the usage of the database.

- It is best if the people using the database look at the contact information table before looking at other parts of the database. This will help whoever wants to use or view the database in understand who is who with regards to artists and customers.
- For security, artists will be able to manipulate a lot of the tables, but not all of them. Leave this to the database administrator. If a user wants something changed they cannot do themselves, then make them ask the database admin.

## Known Problems

**Report “Total Income for Each Artist (with discounts)”**

- Everything in this report is correct except for the income for Artist_ID 1. The price somehow increases by over half the not discounted price. There is nothing happening here that would effect this outcome, and I cannot figure it out. The other income prices are correct.

## Future Enhancements

In the future, I would like to see the following improvements to the current database.

- A way to add or update discounts to customer's profiles.
- Perhaps a table for different types of discounts, for example a friend or family discount category.