|  |
| --- |
|  |
| Commissions Database |
| CMPT 308: Design Project |

|  |
| --- |
| Erina Caferra  4-20-2016 |

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,

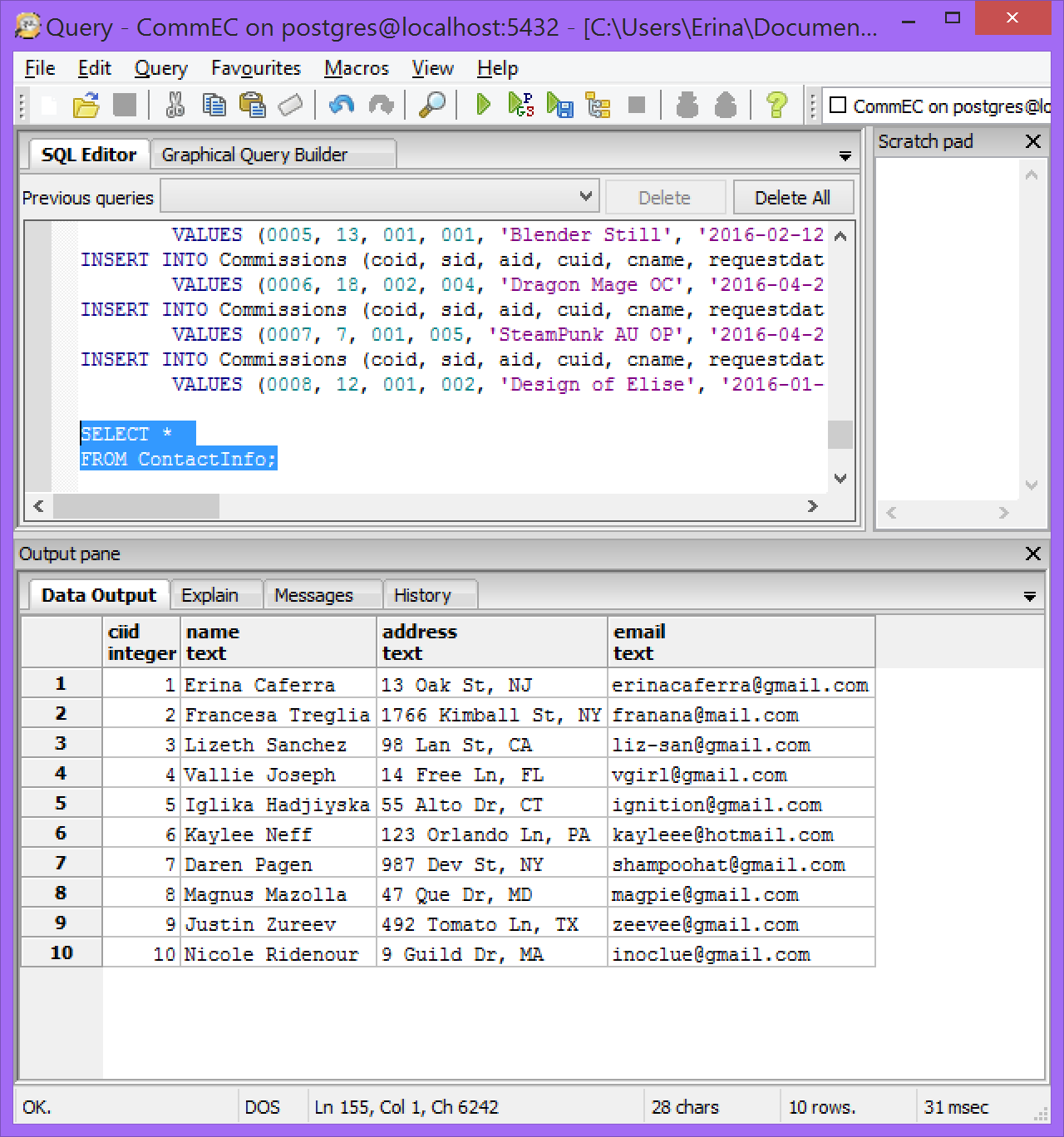
PRIMARY KEY(ciid)

);

**Functional Dependencies:**

CIID 🡪 name, address, email

**Sample Data:**



**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),

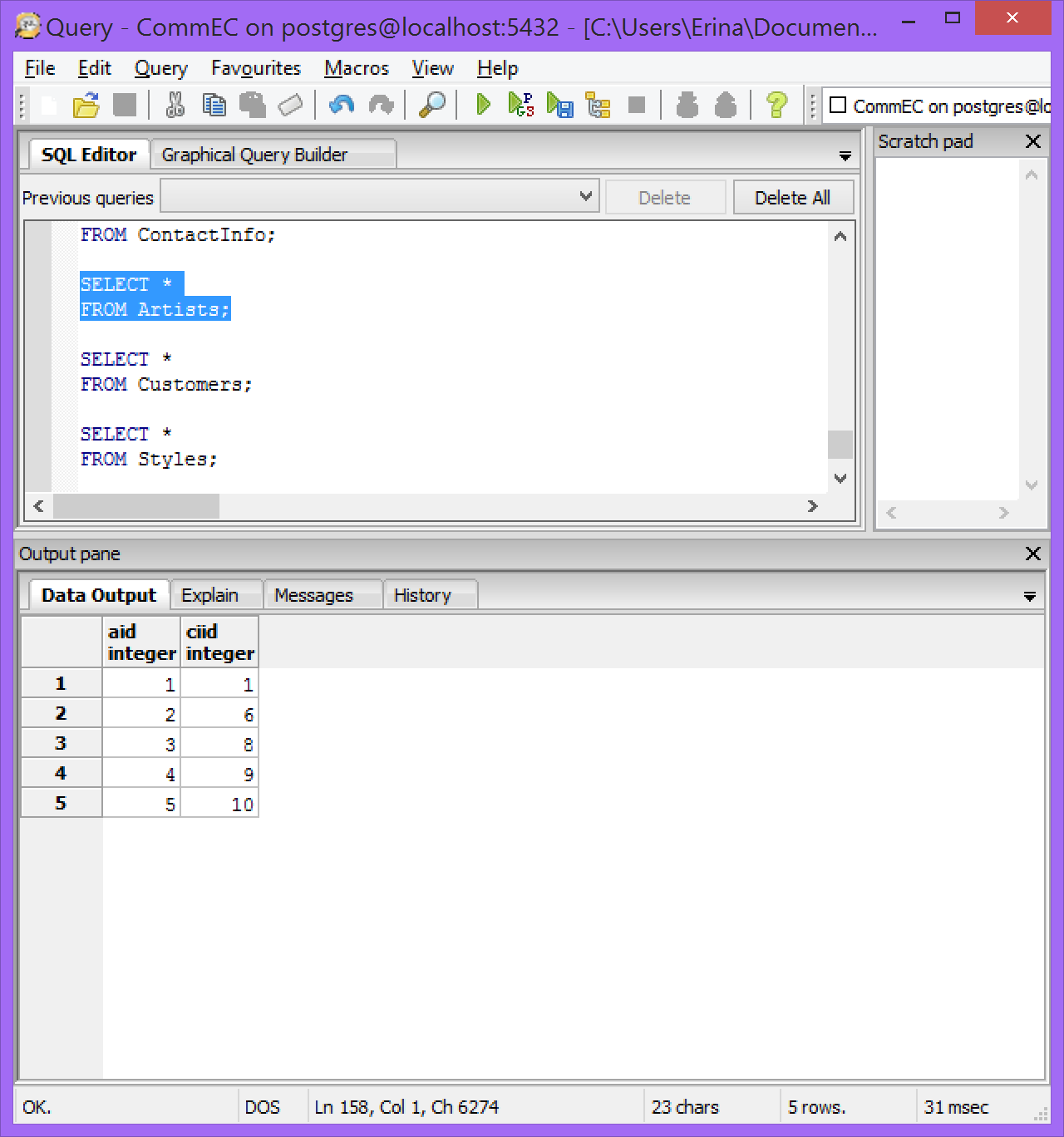
PRIMARY KEY(aid)

);

**Functional Dependencies:**

AID 🡪 CIID

**Sample Data:**



**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,

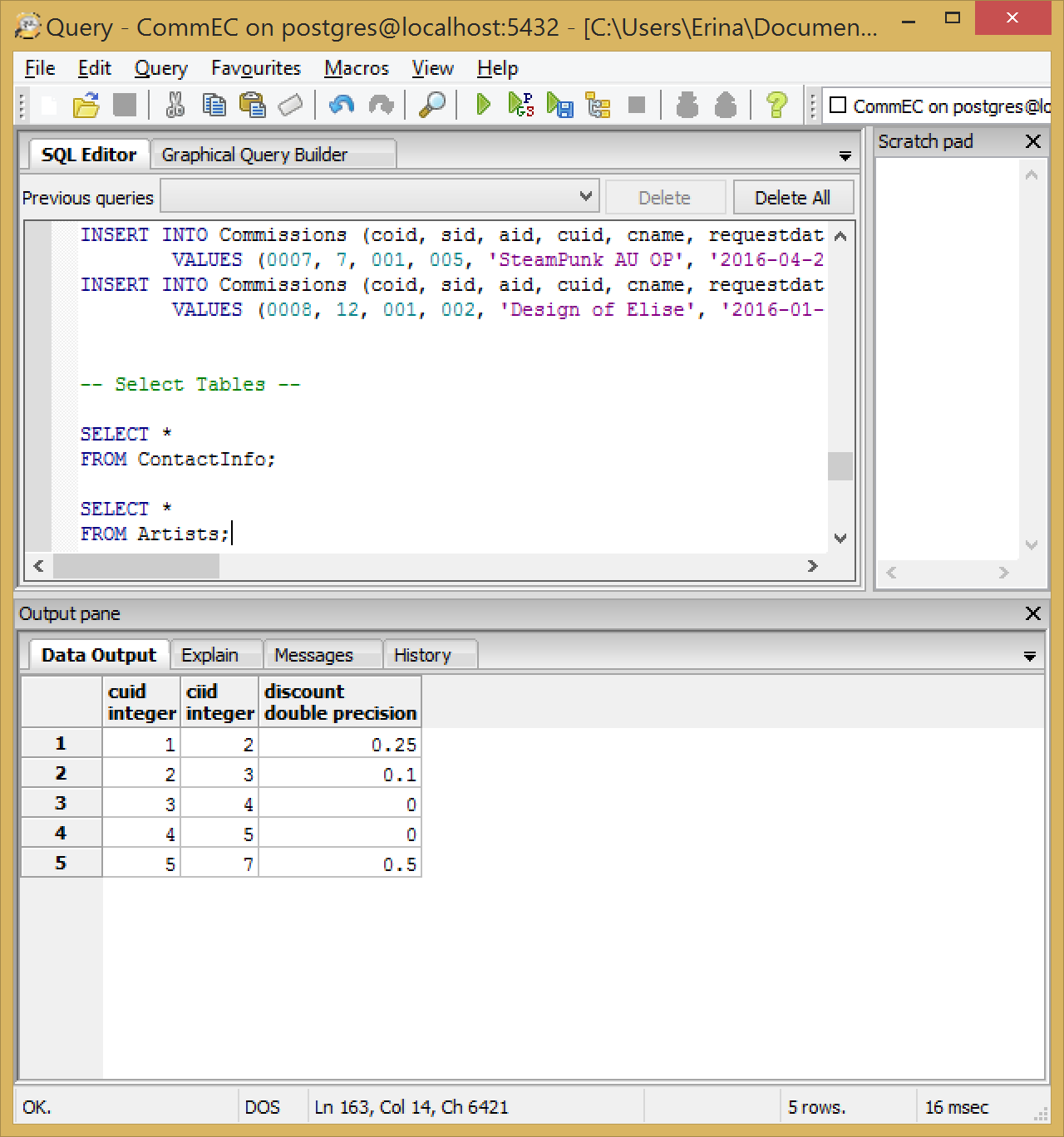
PRIMARY KEY(cuid)

);

**Functional Dependencies:**

CUID 🡪 CIID, Discount

**Sample Data:**



**Styles**

**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,

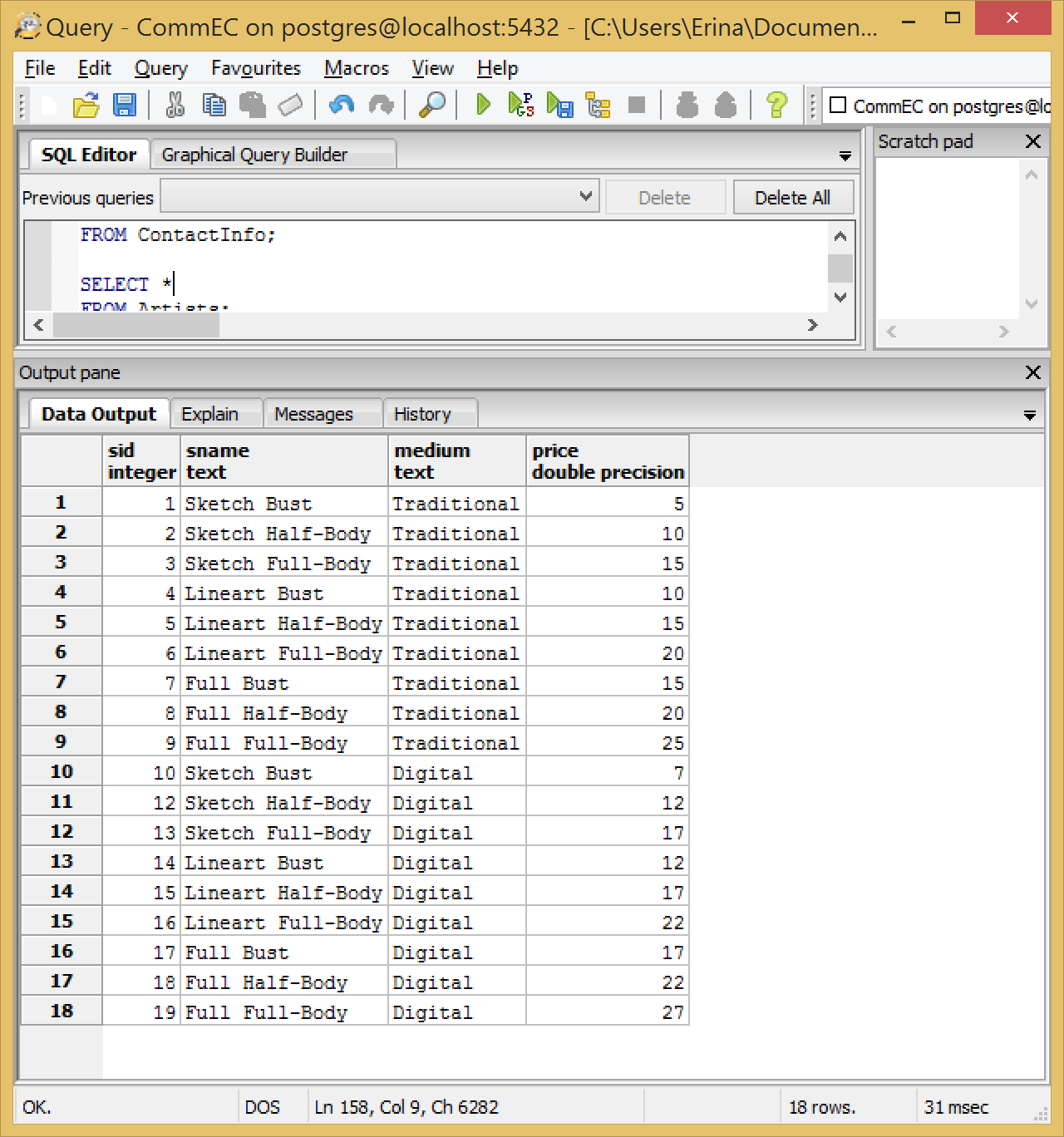
PRIMARY KEY(SID)

);

**Functional Dependencies:**

CUID 🡪 CIID

**Sample Data:**



**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,

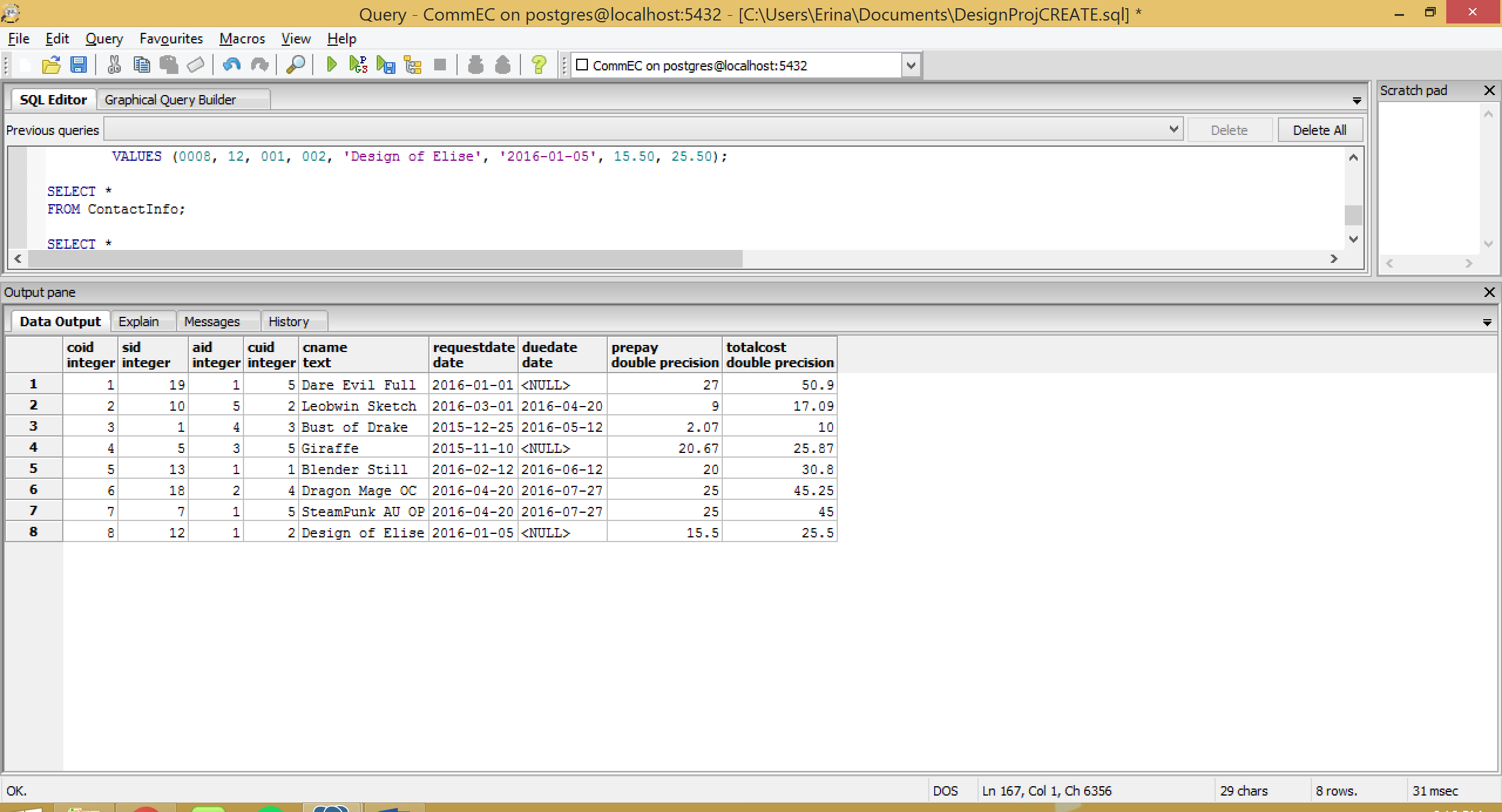
PRIMARY KEY(coid)

);

**Functional Dependencies:**

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

**Sample Data:**



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

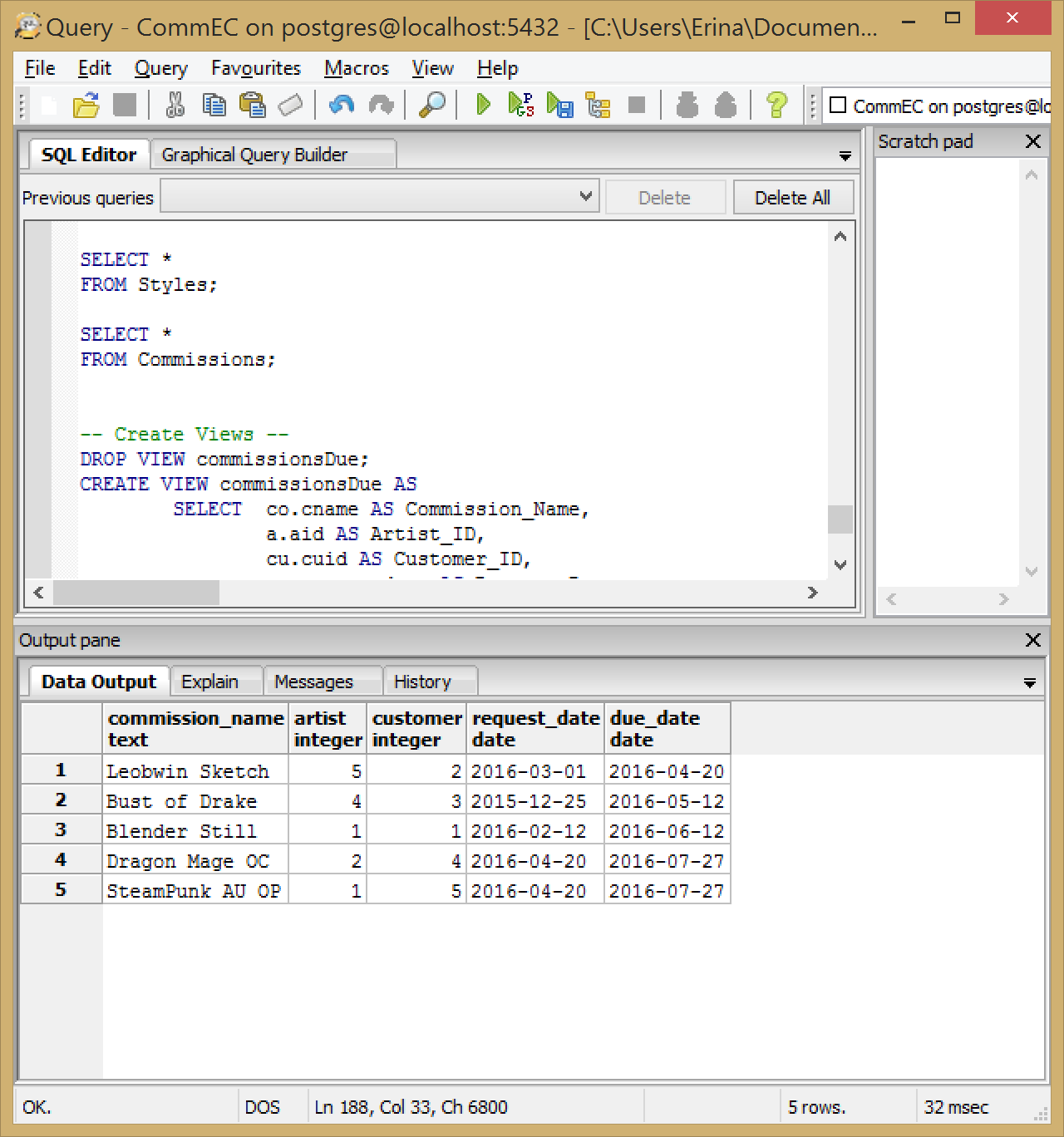
WHERE co.duedate IS NOT NULL

AND a.aid = co.aid

AND cu.cuid = co.cuid

ORDER BY co.duedate ASC;

**Sample Data:**



**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,

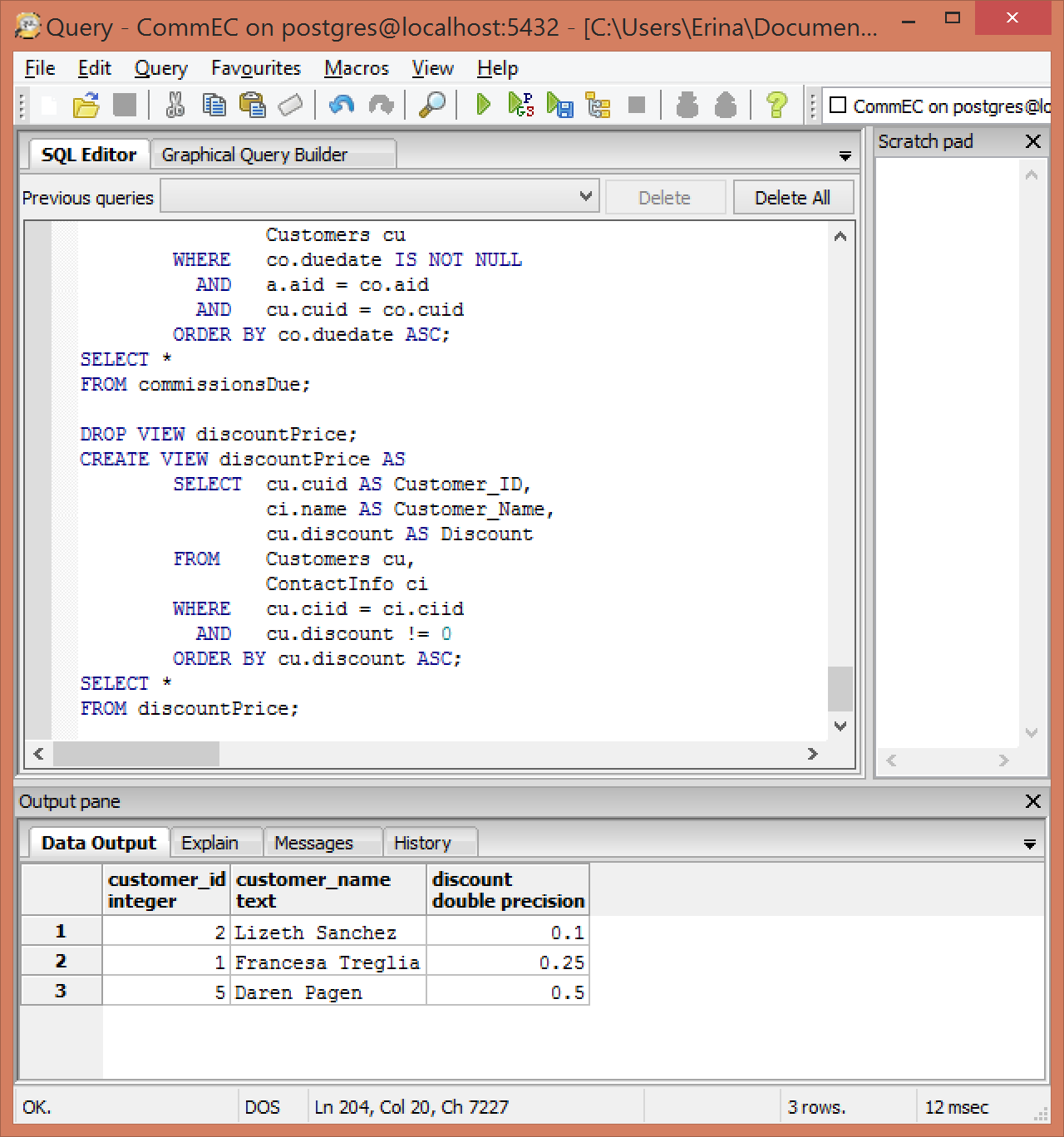
ContactInfo ci

WHERE cu.ciid = ci.ciid

AND cu.discount != 0

ORDER BY cu.discount ASC;

**Sample Data:**



Reports

**Name**

**Purpose:**

**Query:**

**Sample:**

Stored Procedures

Triggers

Security

Implementation Notes

Known Problems

Future Enhancements