DBMS Project Report

PES University

Database Management Systems

UE18CS252

Submitted By

|  |  |
| --- | --- |
| PES2201800156 | ESHAN NAIK |

|  |
| --- |
| An art gallery management system stores the details of paintings, photographs and sculptures in respective tables. It also stores details about the artists, the buyers and the sales each having their own tables. We use another table called artworks which stores the details of the art and the artists.  This system is used for helping a person buy any an art piece for himself or as a gift for someone. This system tells what and all art pieces are available for sale and also gives information on the art piece and the artist who made the piece.  Hence the system has a total of 7 tables with 4 1: N relations in total and a 1 M: N relationship.  This database has all 2NF and 3NF tables.  We used to create table for creating tables for the database and alter table for adding check constraints for the table.  The triggers where created around the selling price of an item and the usual price charged by that artist using create trigger.  There are 2 aggregate queries used, 1 correlated-nested query, 1 query that is both aggregate and correlated-nest and 2 join queries using left join and right join.  We conclude by saying this database is used to store information about an art gallery which can help a person buy art pieces and help the art gallery keep a track of the purchases. |

CONTENTS

[Introduction 2](#_Toc41456463)

[Data Model 2](#_Toc41456464)

[FD and Normalization 5](#_Toc41456465)

[DDL 8](#_Toc41456466)

[Triggers 10](#_Toc41456467)

[SQL Queries 11](#_Toc41456468)

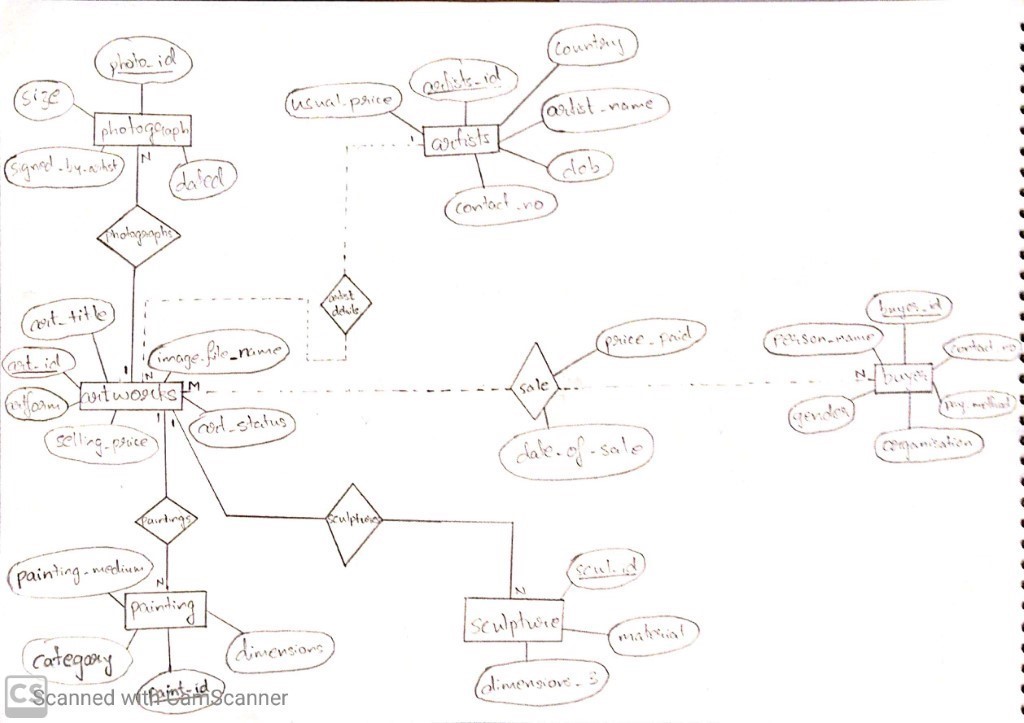
[Conclusion 12](#_Toc41456469)

# Introduction

* This system is used to maintain the market of an art gallery where people give their details which is stored in the table called buyer.
* The details of the artwork which the buyer would want to buy will be stored in a table called the artworks table. This table functions like the art gallery.
* The different forms of artwork each have their own tables. The different forms of artwork are painting, sculpture and photograph.
* Paintings have its details stored in a table called painting.
* Sculptures have its details stored in another table called sculpture.
* Finally, photographs have their details store in another table called photographs.
* The details of the artists who made the art are stored in a table called artists.
* The artwork is sold to a buyer hence here there is an M: N relationship called sale.

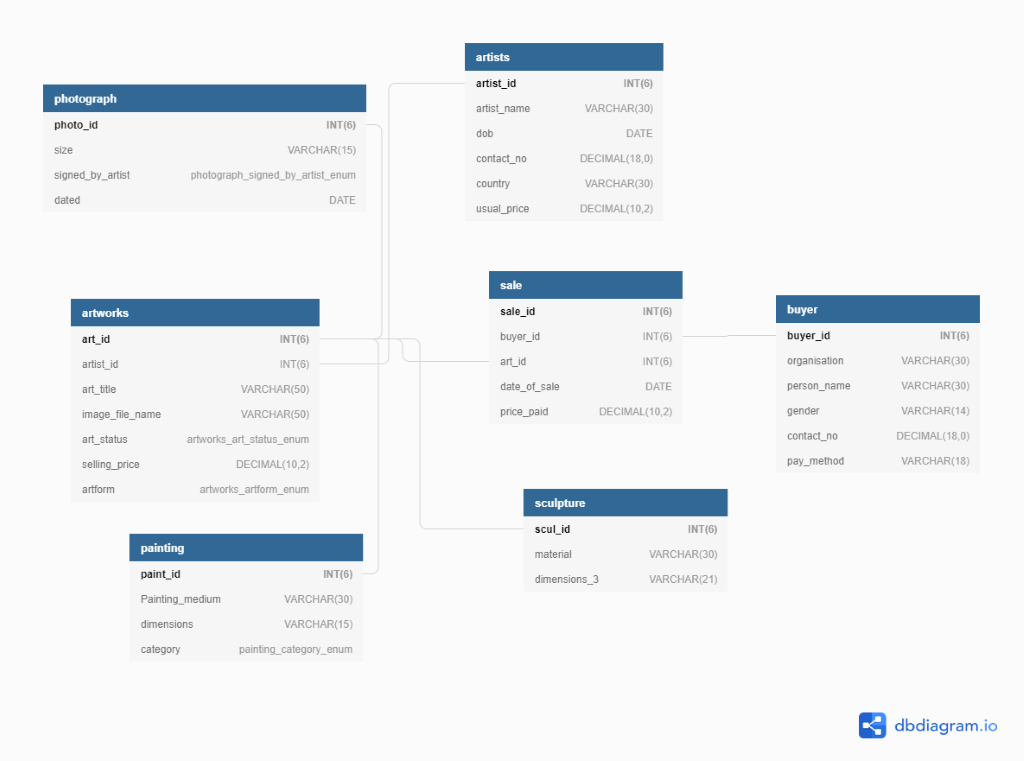
# Data Model

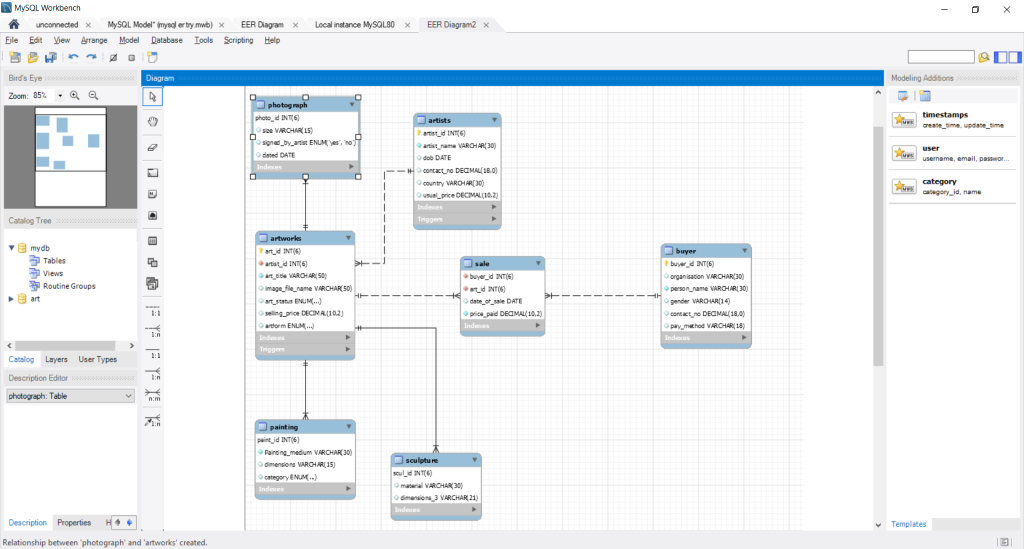
**Given below is the er diagram -**



**Brief description of the ER diagram -**

* Artworks has attributes art\_id, art\_title , image\_file\_name, art\_status, selling\_price and artform, where art\_id is the primary key.
* Artists has attributes artist\_id, artist\_name, dob, contact\_no, country, usual\_price, where artist\_id is the primary key.
* Buyer has attributes buyer\_id, organisation, person\_name, gender, contact\_no, pay\_method, where buyer\_id I a primary key.
* Sale has attributes price\_paid, date\_of\_sale, which is a M: N relationship between artworks and buyer.
* Photographs has attributed photo\_id, size,signed\_by\_artist, dated,where photo\_id is the primary key.
* Painting has attributes painting\_medium, category, paint\_id, dimensions, where paint\_id is a primary key.
* Sculpture has attributes scul\_id, material, dimensions\_3.
* The relationship from artworks to painting,sculpture and painting are all 1: N as there can be multiply paintings, sculptures, photographs but only one art gallery. It is an identifying relationship.
* The relationship between art gallery and buyer is a M: N relationship as multiple buyers can buy multiple art pieces from the same art gallery. and the relation is called sale. It is a non-identifying relationship.
* Sale stores the details of the purchase made by the buyer.
* The relation between artists and artworks is a 1: N relationship as there will be only a single artist who can give many art pieces to the art gallery. It is a non-identifying relation.





# FD and Normalization

artists(artist\_id,artist\_name,dob,contact\_no,country,usual\_price)

Functional dependencies:

artist\_id -> artist\_name

artist\_id -> dob

artist\_id -> contact\_no

artist\_id -> country

artist\_id -> usual\_price

Here artist\_id is a candidate key and is the only candidate key.

Hence the relation is 2NF as there are no partial dependencies.

The relation is 3NF as there are no transition dependencies.

artworks(art\_id,art\_title,artform,selling\_price,art\_status, image\_file\_name)

Functional dependencies:

art\_id -> art\_title

art\_id -> artform

art\_id -> selling\_price

art\_id -> art\_status

art\_id -> image\_file\_name

Here art\_id is a candidate key and is the only candidate key.

Hence the relation is 2NF as there are no partial dependencies.

The relation is 3NF as there are no transition dependencies.

buyer(buyer\_id,organisation, person\_name,gender,contact\_no,pay\_method)

Functional dependencies:

buyer\_id -> organisation

buyer\_id -> person\_name

buyer\_id -> gender

buyer\_id -> contact\_no

buyer\_id -> pay\_method

Here buyer\_id is a candidate key and is the only candidate key.

Hence the relation is 2NF as there are no partial dependencies.

The relation is 3NF as there are no transition dependencies.

painting(paint\_id,painting\_medium,dimensions,category)

Functional dependencies:

paint\_id -> painting\_medium

paint\_id -> dimensions

paint\_id -> category

Here paint\_id is the only candidate key.

The relation is 2NF as there are no partial dependencies

The relations are 3NF as there are no transition dependencies

photograph(photo\_id,size,signed\_by\_artist,dated)

Functional dependencies:

photo\_id -> size

photo\_id -> signed\_by\_artist

photo\_id -> dated

Here photo\_id is a candidate key and the only candidate key.

Hence the relation is 2NF as there are no partial dependencies.

The relation is also 3NF as there are no transitive dependencies.

sale(buyer\_id,art\_id,date\_of\_sale,price\_paid)

Functional dependencies:

{buyer\_id,art\_id} -> date\_of\_sale

{buyer\_id,art\_id} -> price\_paid

Here {sale\_id,buyer\_id,art\_id} is a candidate key and is the only candidate key.

Hence the relation is 2NF as there are no partial dependencies.

The relation is 3NF as there are no transition dependencies.

sculpture(scul\_id,material,dimensions\_3)

Functional depencies:

scul\_id -> material

scul\_id -> dimensions\_3

Here scul\_id is a candidate key and the only candidate key.

Hence the relation is 2NF as there are no partial dependencies.

The relation is also 3NF as there are no transitive dependencies.

**Discussion on which normal form will be violated if 2 relations are combined**

Ex 1: When we combine tables sale and buyer then

buyer\_id -> organisation

buyer\_id -> person\_name

buyer\_id -> gender

buyer\_id -> contact\_no

buyer\_id -> pay\_method

{buyer\_id,art\_id} -> date\_of\_sale

{buyer\_id,art\_id} -> price\_paid

The closure of buyer\_id = { organisation,person\_name, gender,contact\_no,pay\_method}, that is, buyer\_id is not the candidate key in this combined table.

Now the closure of {buyer\_id,art\_id} = { organisation,person\_name, gender, contact\_no, pay\_method, price\_paid, date\_of\_sale}. This tells us that {buyer\_id,art\_id} is the candidate key.

This show that when we combine the tables sale and buyer then there will be a 2NF violation because of partial dependencies like buyer\_id ->organisation and buyer\_id ->person\_name.

Ex 2: If we take the tables artwork and sale

art\_id -> art\_title

art\_id -> artform

art\_id -> selling\_price

art\_id -> art\_status

art\_id -> image\_file\_name

{buyer\_id,art\_id} -> date\_of\_sale

{buyer\_id,art\_id} -> price\_paid

The closure of art\_id ={art\_title, artform, selling\_price, art\_status, image\_file\_name}. This tells us that art\_id is not a candidate key for the combined table.

Now the closure of {buyer\_id,art\_id} = { art\_title, artform, selling\_price, art\_status, image\_file\_name, price\_paid, date\_of\_sale}. This shows that {buyer\_id,sale\_id} is the candidate key as it has all the attributes of the combined table.

Now we can say that this violates 2NF as there are partial dependencies like art\_id -> art\_title and art\_id -> artform.

Ex 3: If we take the tables sculpture and artworks

art\_id -> art\_title

art\_id -> artform

art\_id -> selling\_price

art\_id -> art\_status

art\_id -> image\_file\_name

art\_id -> scul\_id

scul\_id -> material

scul\_id -> dimensions\_3

The closure of scul\_id = {material, dimensions\_3}. This tells us that scul\_id is not a candidate key for the combined table.

Now the closure of art\_id ={art\_title, artform, selling\_price, art\_status, image\_file\_name, dimensions\_3, scul\_id, dimensions\_3, material}. This shows that art\_id is the candidate key as it has all the attributes of the combined table.

Now we can say that this violates 3NF as there are transition dependencies like art\_id -> scul\_id and scul\_id -> dimensions\_3, hence art\_id -> dimensions\_3.

# DDL

**Creating Tables**

CREATE TABLE IF NOT EXISTS `art`.`artists` (

`artist\_id` INT(6) NOT NULL AUTO\_INCREMENT,

`artist\_name` VARCHAR(30) NOT NULL,

`dob` DATE NULL DEFAULT NULL,

`contact\_no` DECIMAL(18,0) NULL DEFAULT NULL,

`country` VARCHAR(30) NULL DEFAULT NULL,

`usual\_price` DECIMAL(10,2) NULL DEFAULT NULL,

PRIMARY KEY (`artist\_id`))

CREATE TABLE IF NOT EXISTS `art`.`artworks` (

`art\_id` INT(6) NOT NULL,

`artist\_id` INT(6) NOT NULL,

`art\_title` VARCHAR(50) NOT NULL,

`image\_file\_name` VARCHAR(50) NULL DEFAULT NULL,

`art\_status` ENUM('available', 'sold', 'reserved', 'consigned out') NULL DEFAULT 'available',

`selling\_price` DECIMAL(10,2) NULL DEFAULT NULL,

`artform` ENUM('painting', 'photograph', 'sculpture') NULL DEFAULT 'painting',

PRIMARY KEY (`art\_id`),

INDEX `artist\_id` (`artist\_id` ASC) VISIBLE,

CONSTRAINT `artworks\_ibfk\_1`

FOREIGN KEY (`artist\_id`)

REFERENCES `art`.`artists` (`artist\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE)

CREATE TABLE IF NOT EXISTS `art`.`buyer` (

`buyer\_id` INT(6) NOT NULL AUTO\_INCREMENT,

`organisation` VARCHAR(30) NULL DEFAULT NULL,

`person\_name` VARCHAR(30) NOT NULL,

`gender` VARCHAR(14) NULL DEFAULT NULL,

`contact\_no` DECIMAL(18,0) NULL DEFAULT NULL,

`pay\_method` VARCHAR(18) NULL DEFAULT NULL,

PRIMARY KEY (`buyer\_id`))

CREATE TABLE IF NOT EXISTS `art`.`painting` (

`paint\_id` INT(6) NOT NULL AUTO\_INCREMENT,

`Painting\_medium` VARCHAR(30) NOT NULL,

`dimensions` VARCHAR(15) NULL DEFAULT NULL,

`category` ENUM('landscape', 'portrait', 'abstract', 'still-life') NULL DEFAULT 'landscape',

PRIMARY KEY (`paint\_id`),

CONSTRAINT `painting\_ibfk\_1`

FOREIGN KEY (`paint\_id`)

REFERENCES `art`.`artworks` (`art\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE)

CREATE TABLE IF NOT EXISTS `art`.`photograph` (

`photo\_id` INT(6) NOT NULL,

`size` VARCHAR(15) NULL DEFAULT NULL,

`signed\_by\_artist` ENUM('yes', 'no') NULL DEFAULT 'yes',

`dated` DATE NULL DEFAULT NULL,

PRIMARY KEY (`photo\_id`),

CONSTRAINT `photograph\_ibfk\_1`

FOREIGN KEY (`photo\_id`)

REFERENCES `art`.`artworks` (`art\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE)

CREATE TABLE IF NOT EXISTS `art`.`sale` (

`buyer\_id` INT(6) NOT NULL,

`art\_id` INT(6) NOT NULL,

`date\_of\_sale` DATE NULL DEFAULT NULL,

`price\_paid` DECIMAL(10,2) NOT NULL,

INDEX `buyer\_id` (`buyer\_id` ASC) VISIBLE,

INDEX `art\_id` (`art\_id` ASC) VISIBLE,

CONSTRAINT `sale\_ibfk\_1`

FOREIGN KEY (`buyer\_id`)

REFERENCES `art`.`buyer` (`buyer\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE,

CONSTRAINT `sale\_ibfk\_2`

FOREIGN KEY (`art\_id`)

REFERENCES `art`.`artworks` (`art\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE)

CREATE TABLE IF NOT EXISTS `art`.`sculpture` (

`scul\_id` INT(6) NOT NULL,

`material` VARCHAR(30) NULL DEFAULT 'stone',

`dimensions\_3` VARCHAR(21) NULL DEFAULT NULL,

PRIMARY KEY (`scul\_id`),

CONSTRAINT `sculpture\_ibfk\_1`

FOREIGN KEY (`scul\_id`)

REFERENCES `art`.`artworks` (`art\_id`)

ON DELETE RESTRICT

ON UPDATE CASCADE)

**CHECK CONSTRAINTS**

ALTER TABLE artists ADD CONSTRAINT age\_restriction CHECK (dob between ‘1920-01-01' AND ‘2002-05-22');

ALTER TABLE buyer ADD CONSTRAINT check\_pay\_method CHECK (pay\_method in (‘Debit Care’, ‘Credit Card’, ‘Cheque’);

# Triggers

We know that the usual price of charged by an artists cannot be a negative value so we can use a trigger to make sure that if the usual price an artist charges is entered as a negative value by mistake we should make it 0, which means the artists does work for free.

USE `art`;

DELIMITER $$

USE `art`$$

CREATE

DEFINER=`root`@`localhost`

TRIGGER `art`.`before\_usual\_price`

BEFORE INSERT ON `art`.`artists`

FOR EACH ROW

BEGIN

IF NEW.usual\_price < 0 THEN SET NEW.usual\_price = '0';

END IF;

END$$

We know that the price of any art piece cannot be negative so we make a trigger to make selling price to 0 if it is inputted as a negative number.

USE `art`$$

CREATE

DEFINER=`root`@`localhost`

TRIGGER `art`.`before\_selling\_price`

BEFORE UPDATE ON `art`.`artworks`

FOR EACH ROW

BEGIN IF NEW.selling\_price < 0 THEN SET NEW.selling\_price = 0;

END IF;

END$$

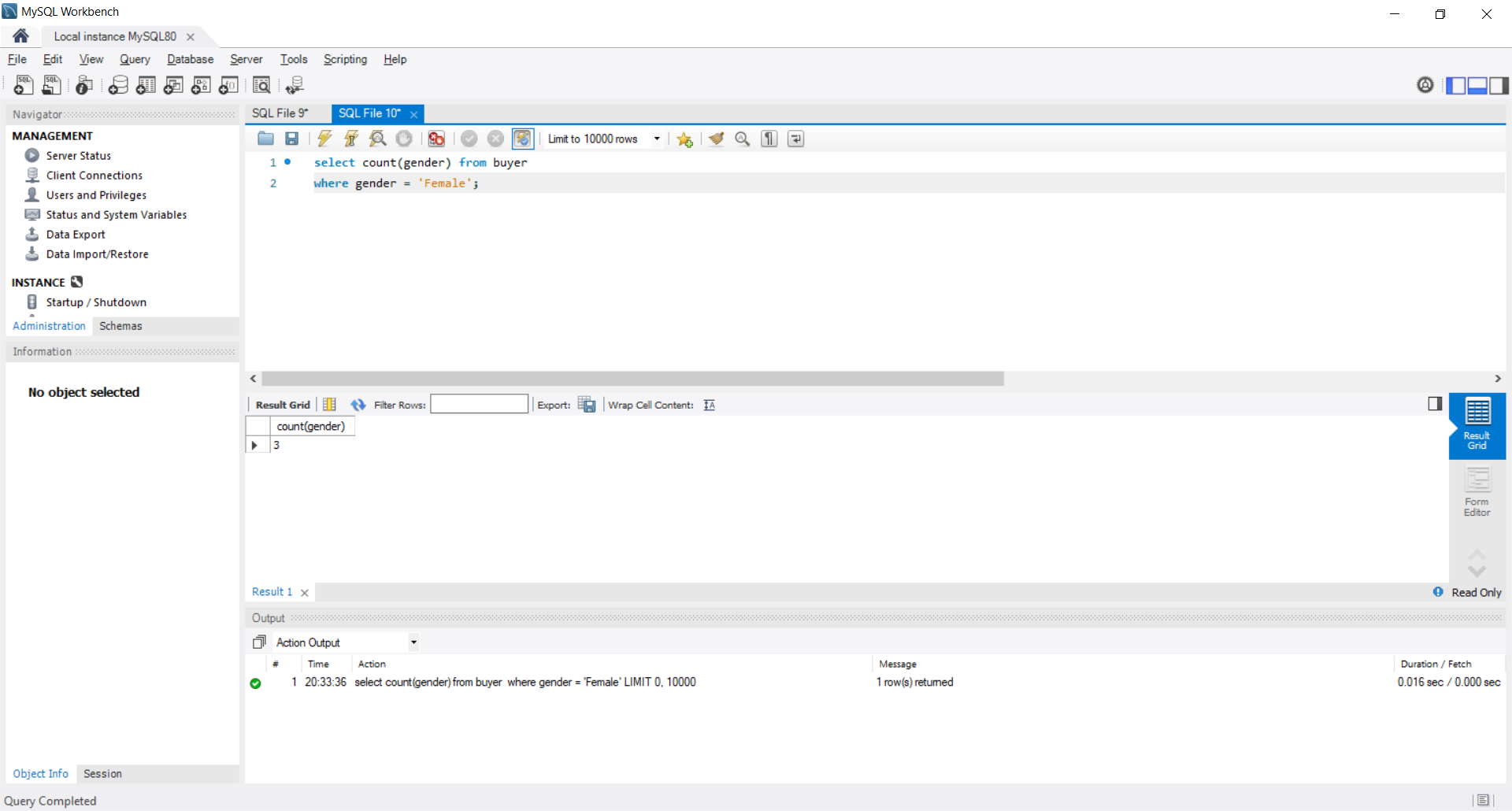
# SQL Queries

**Aggregate Functions**

**SQL query to count the number of female buyers**

select count(gender) from buyer

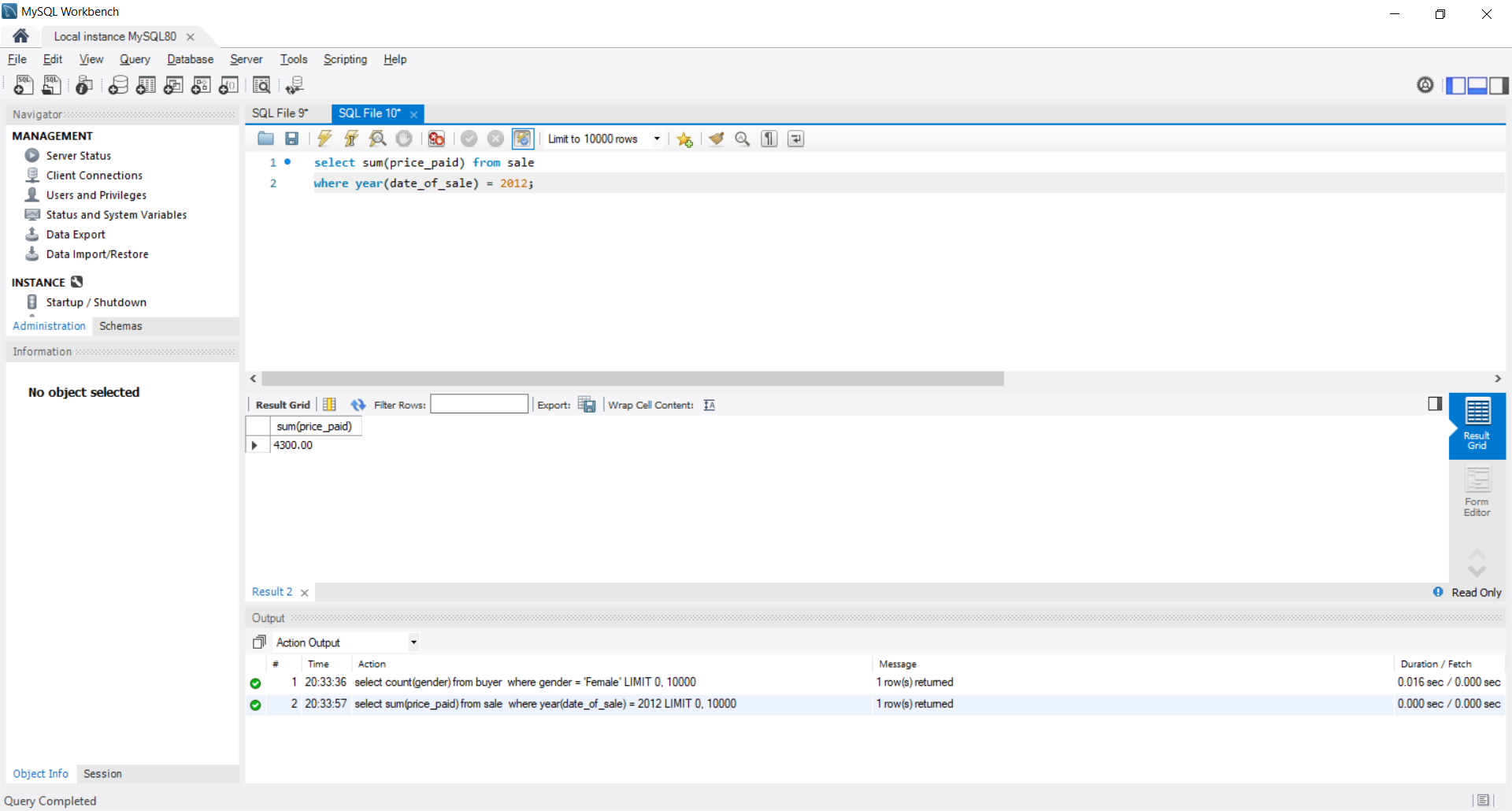
where gender = 'Female';



**SQL query to find the sum of sales in the year 2013**

select sum(price\_paid) from sale

where year(date\_of\_sale) = 2012;



**Join functions**

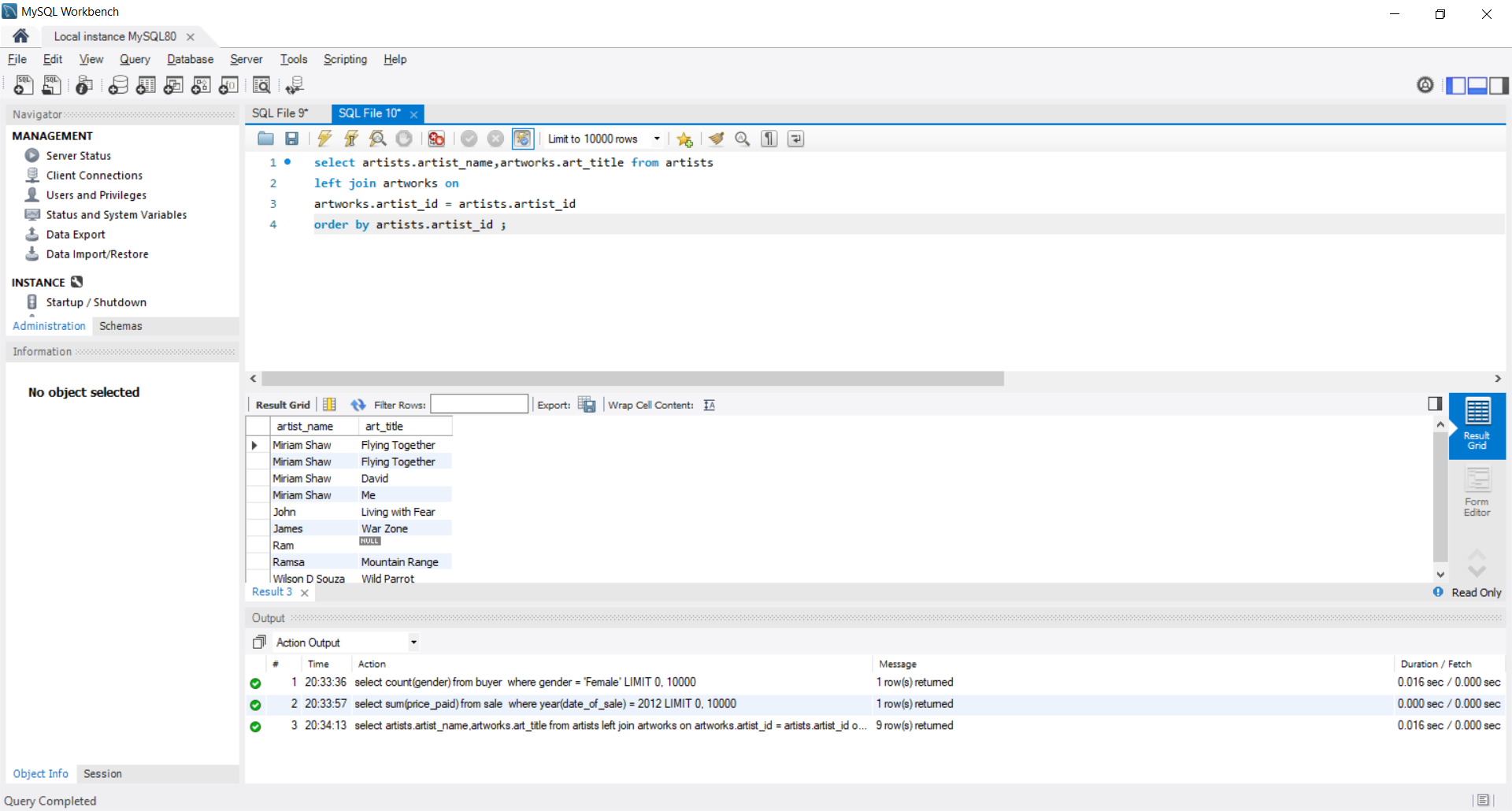
**SQL query for a left join between artist name and the title of his art piece**

select artists.artist\_name,artworks.art\_title from artists

left join artworks on

artworks.artist\_id = artists.artist\_id

order by artists.artist\_id ;



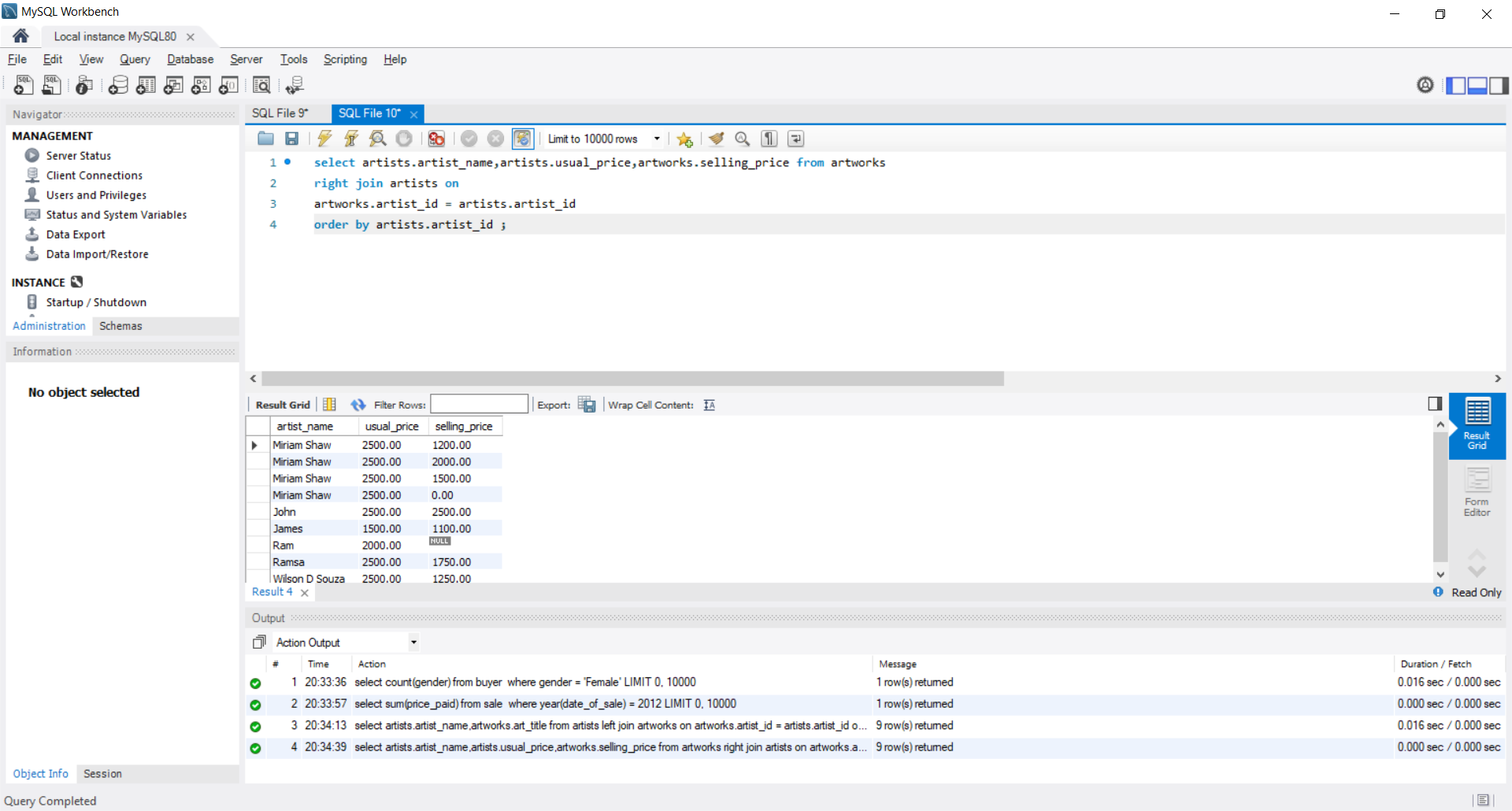
**SQL query for a right join between artist name the usual price he charges and the price at which he sold it**

select artists.artist\_name,artists.usual\_price,artworks.selling\_price from artworks

right join artists on

artworks.artist\_id = artists.artist\_id

order by artists.artist\_id ;



**Correlated Functions**

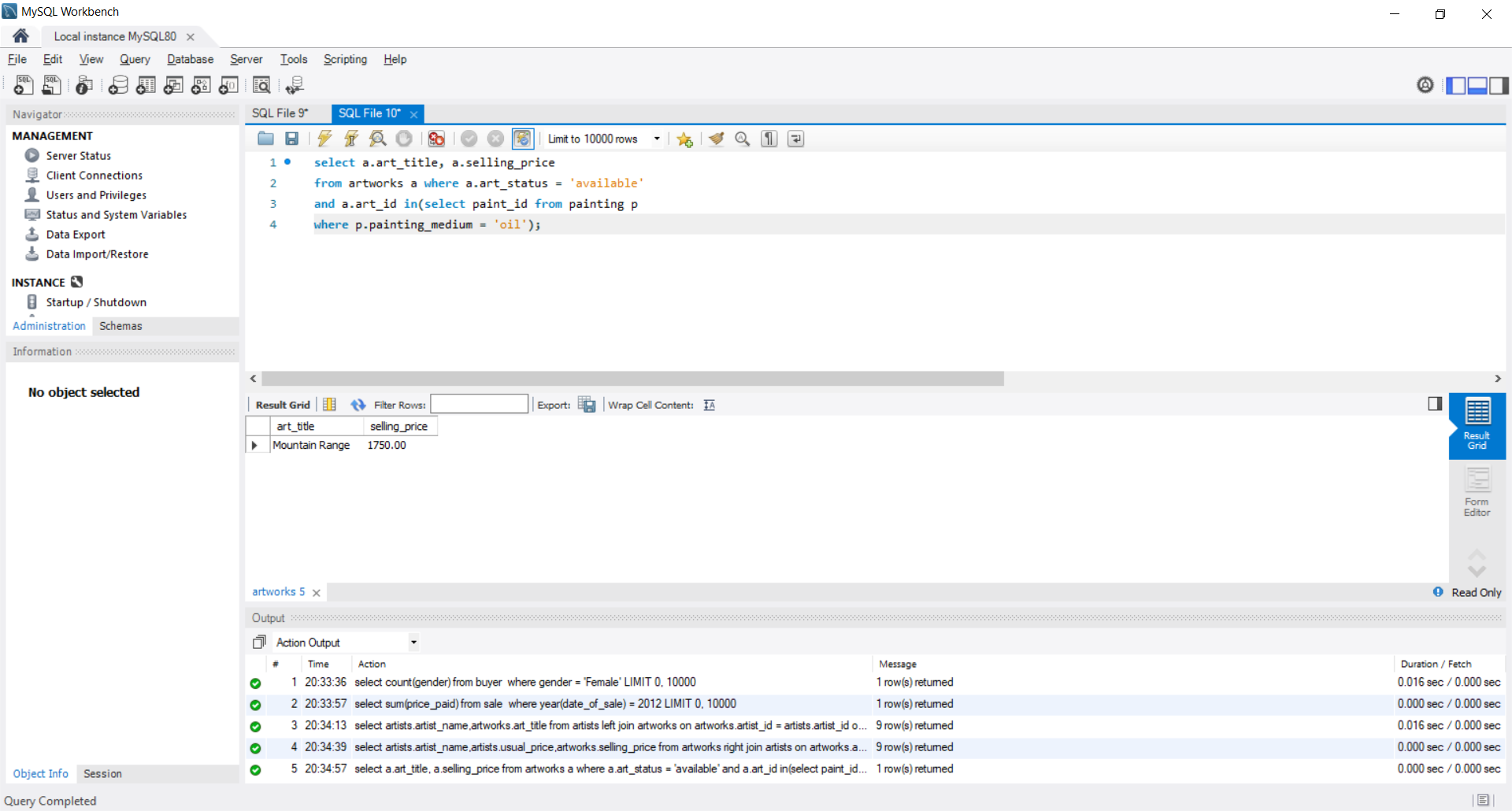
**SQL query which shows the available paintings which are for sale and are made from oil.**

select a.art\_title, a.selling\_price

from artworks a where a.art\_status = 'available'

and a.art\_id in (select paint\_id from painting p

where p.painting\_medium = 'oil');

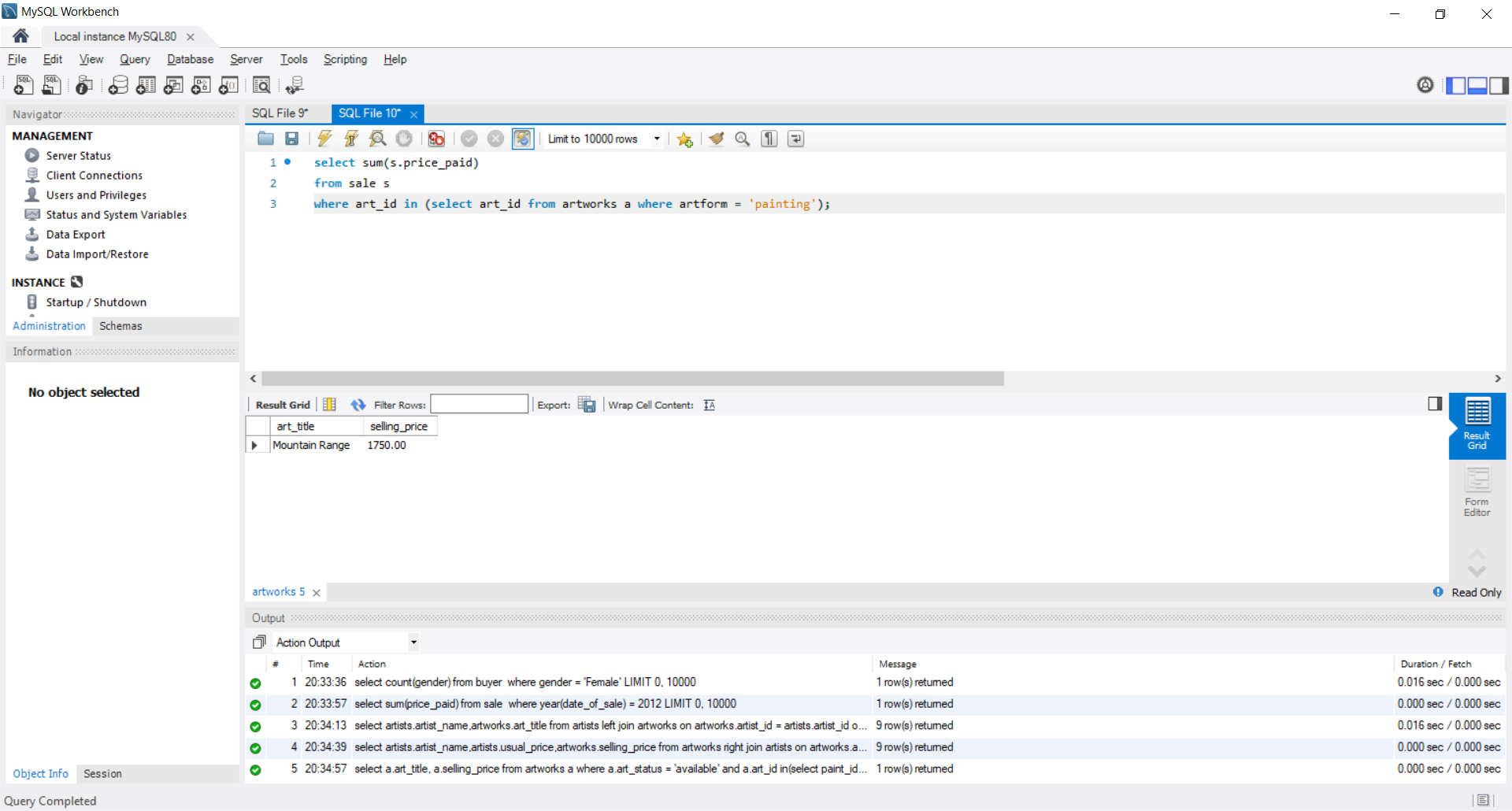


**SQL query to find total sales on painting**

select sum(s.price\_paid)

from sale s

where art\_id in (select art\_id from artworks a where artform = 'painting');



# Conclusion

We can conclude that this database can be used by a person to find artwork he would like to purchase and help the art gallery maintain a database for all the artwork the art gallery is selling.

We use triggers to make sure that selling price of any artwork is not a negative number but instead will be set to 0, that is, it will be sold for free.

The sql queries we use are to find the maximum money made from a single art piece or to count the total number of art pieces for sale. It is also used to find out more details about the sales of the art gallery.

Future Work:

I would like to add more artforms to the art gallery and also make it so the we can deliver the artwork. Increase the amount of info stored about the artworks and artists.