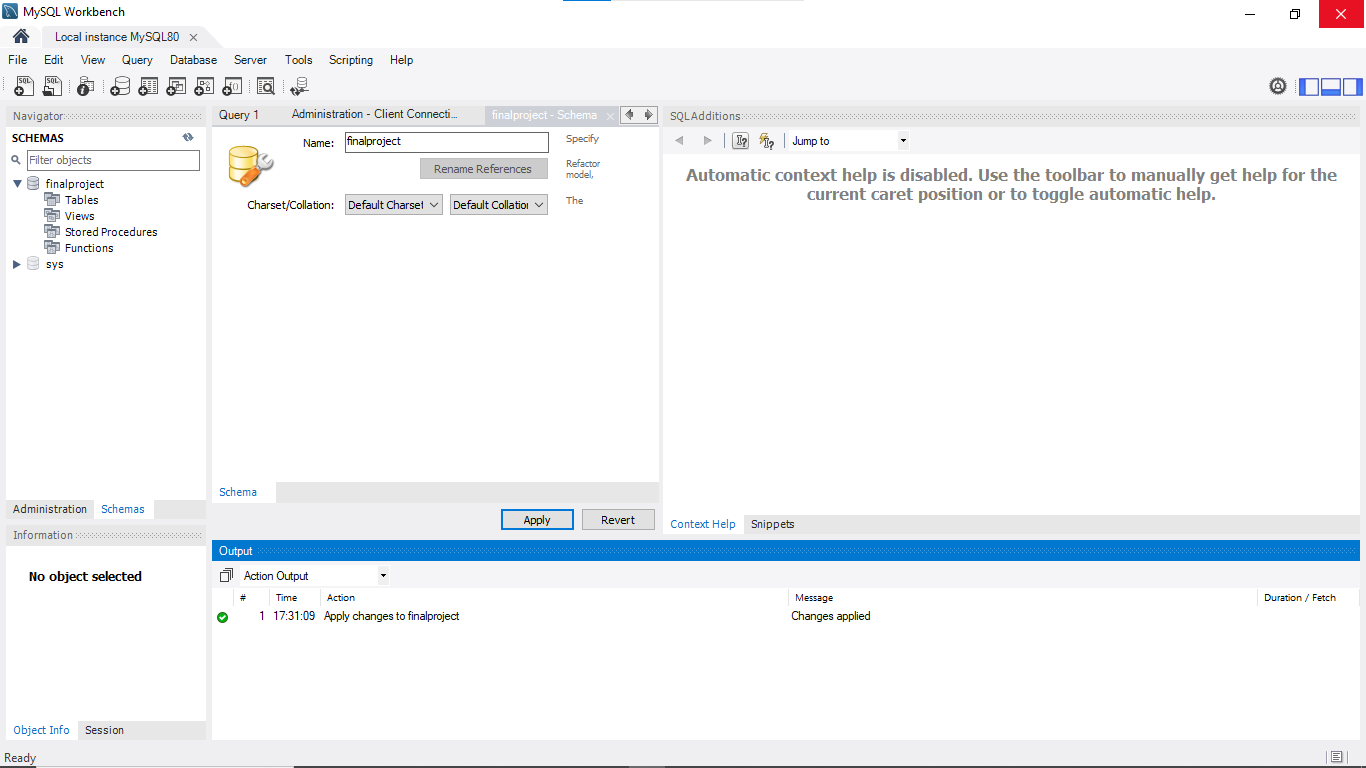
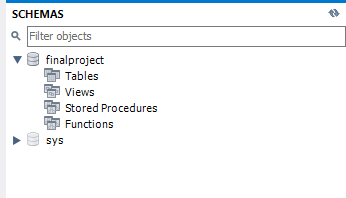
**Prompt 1**





**PROMPT 2:**

USE finalproject;

**-- Create the `users` table to store information about users of the database**

CREATE TABLE users (

userid INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,UNIQUE,

name VARCHAR(255),

username VARCHAR(20),

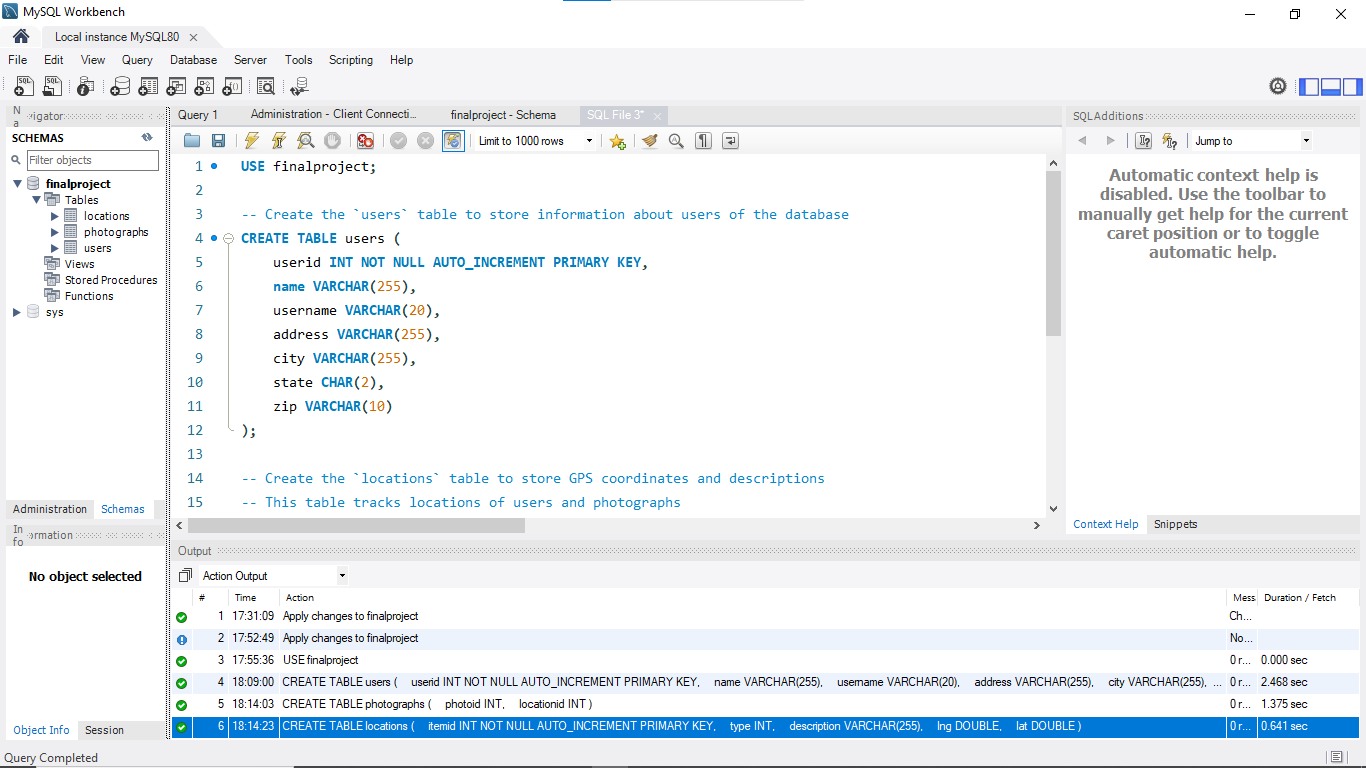
address VARCHAR(255),

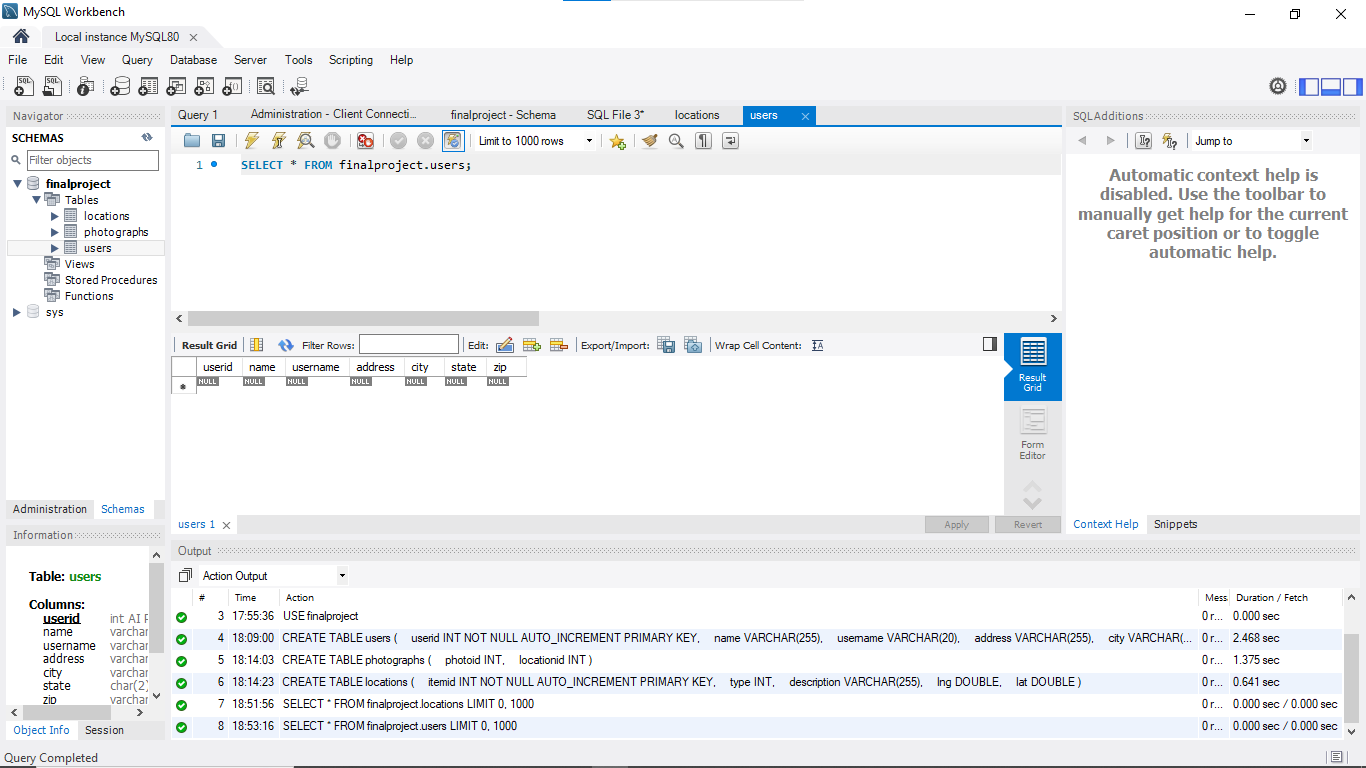
city VARCHAR(255),

state CHAR(2),

zip VARCHAR(10)

);





**-- Create the `locations` table to store GPS coordinates and descriptions**

-- This table tracks locations of users and photographs

CREATE TABLE locations (

itemid INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

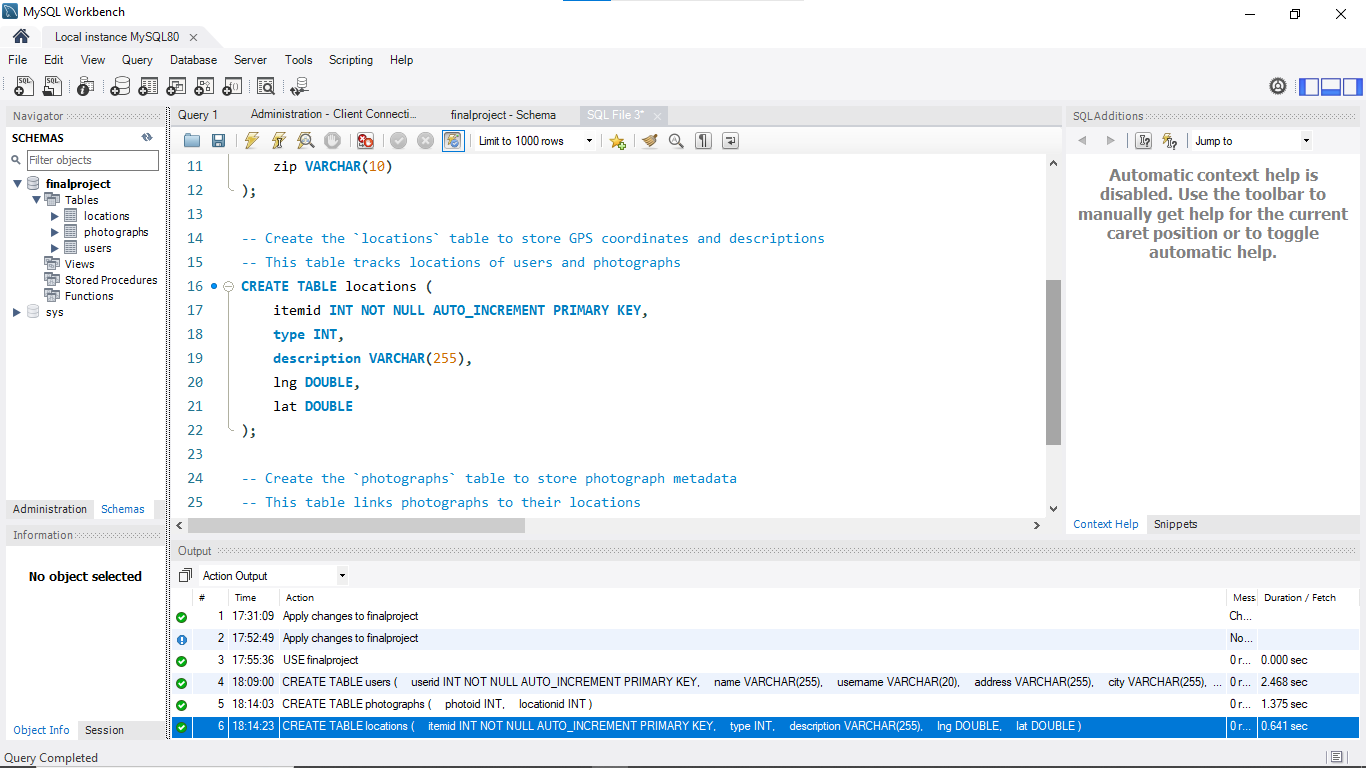
type INT,

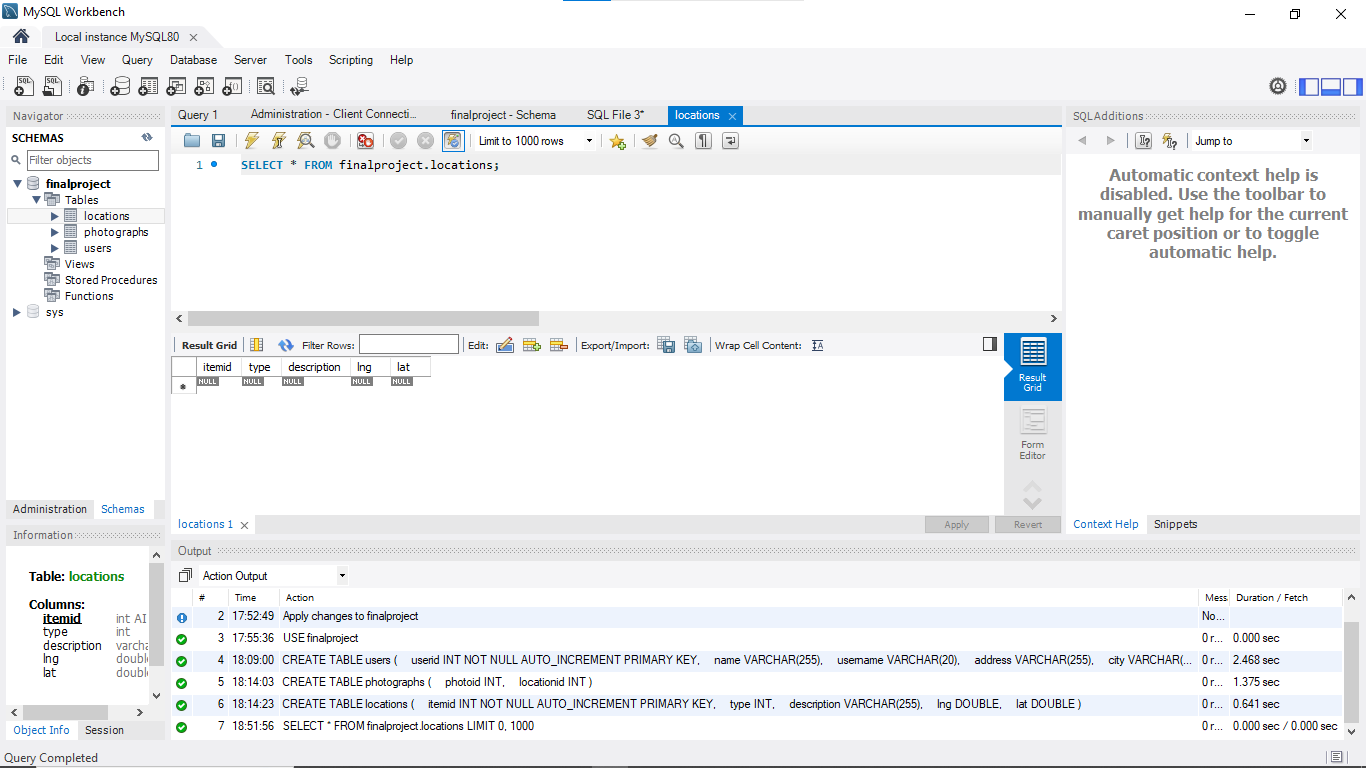
description VARCHAR(255),

lng DOUBLE,

lat DOUBLE

);





**-- Create the `photographs` table to store photograph metadata**

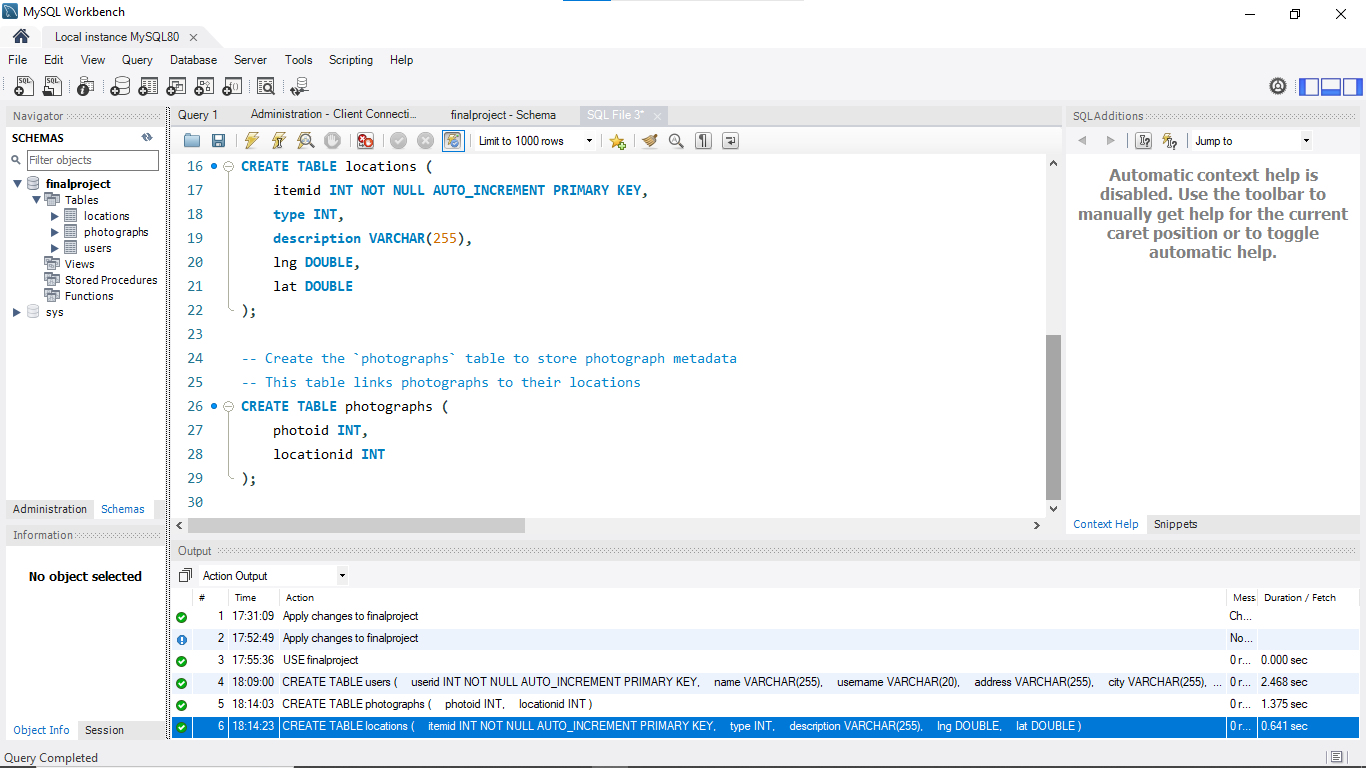
-- This table links photographs to their locations

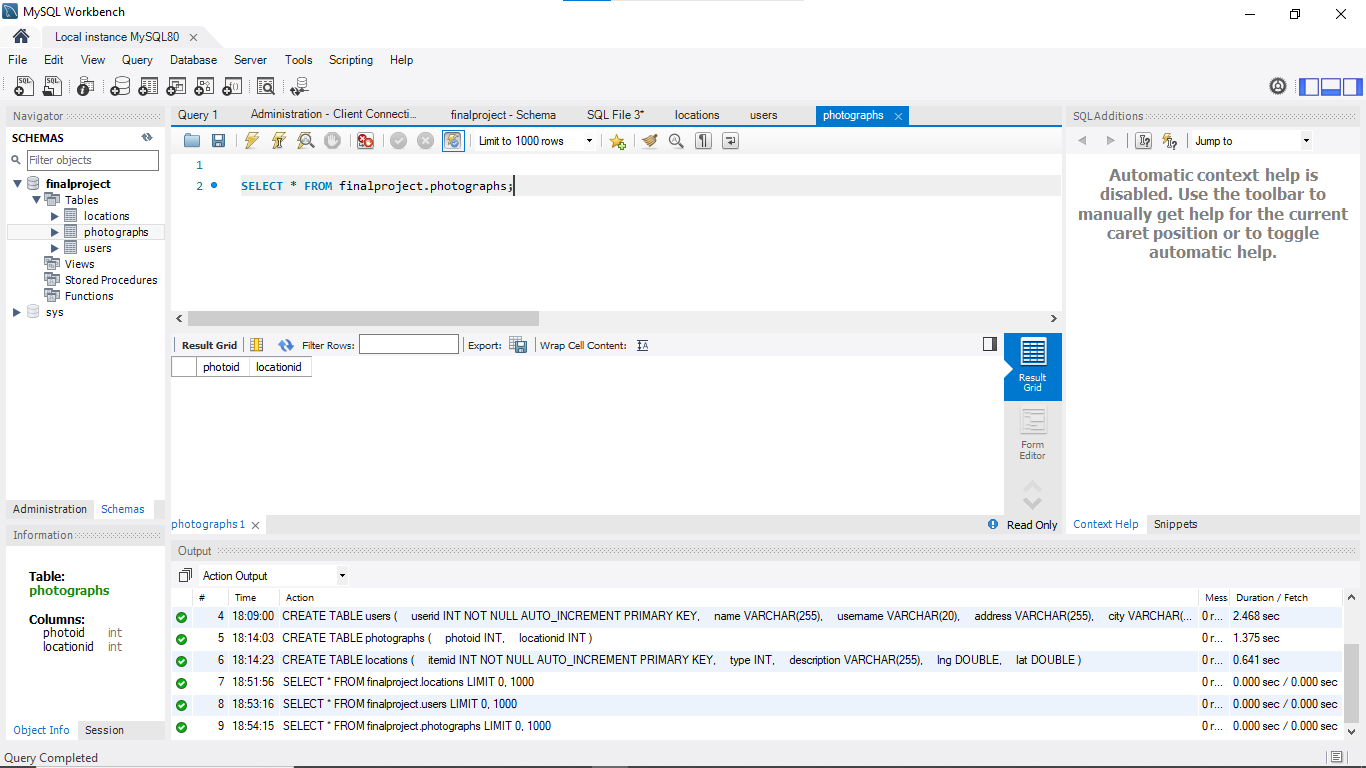
CREATE TABLE photographs (

photoid INT,

locationid INT

);





**PROMPT 3:**

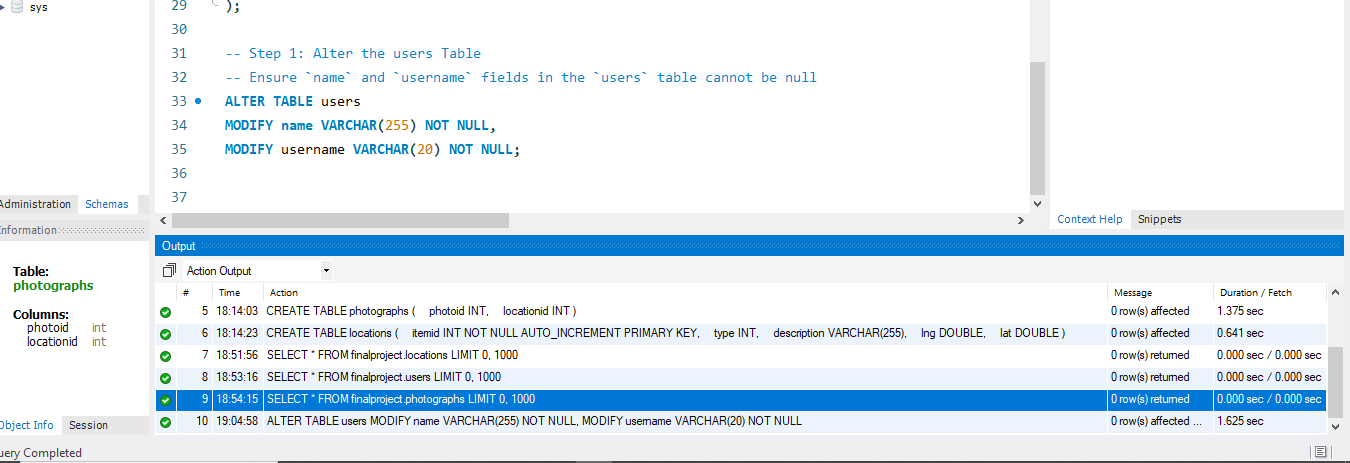
-- Step 1: Alter the users Table

-- Ensure `name` and `username` fields in the `users` table cannot be null

ALTER TABLE users

MODIFY name VARCHAR(255) NOT NULL,

MODIFY username VARCHAR(20) NOT NULL;

****

-- Step 2: Alter the locations Table

-- Ensure `type`, `description`, `lng`, and `lat` fields in the `locations` table cannot be null

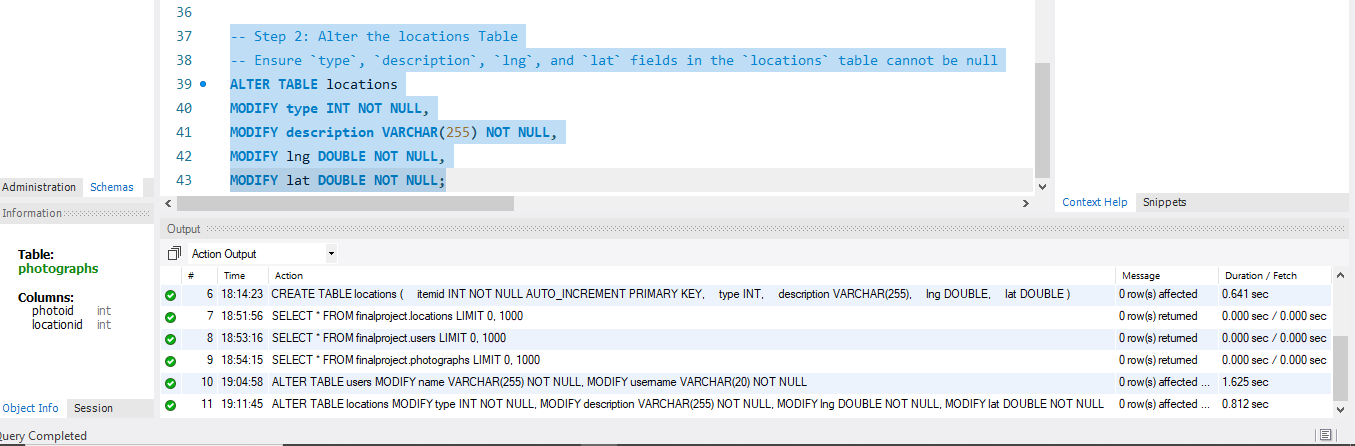
ALTER TABLE locations

MODIFY type INT NOT NULL,

MODIFY description VARCHAR(255) NOT NULL,

MODIFY lng DOUBLE NOT NULL,

MODIFY lat DOUBLE NOT NULL;



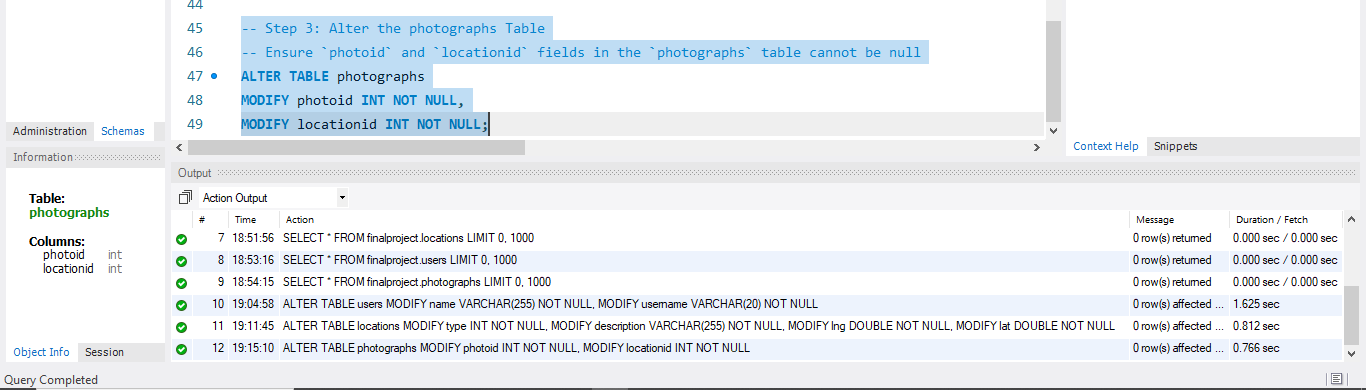
-- Step 3: Alter the photographs Table

-- Ensure `photoid` and `locationid` fields in the `photographs` table cannot be null

ALTER TABLE photographs

MODIFY photoid INT NOT NULL,

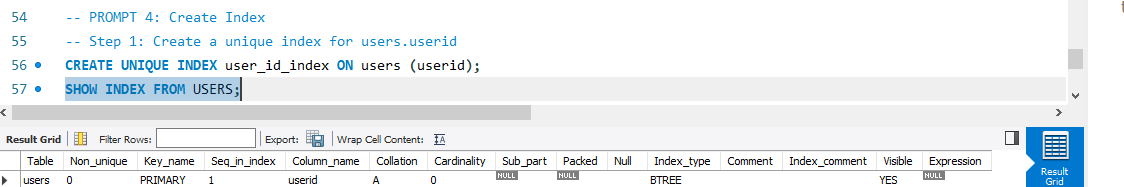
MODIFY locationid INT NOT NULL;



**PROMPT 4: Create Index**

**-- Step 1: Create a unique index for users.userid**

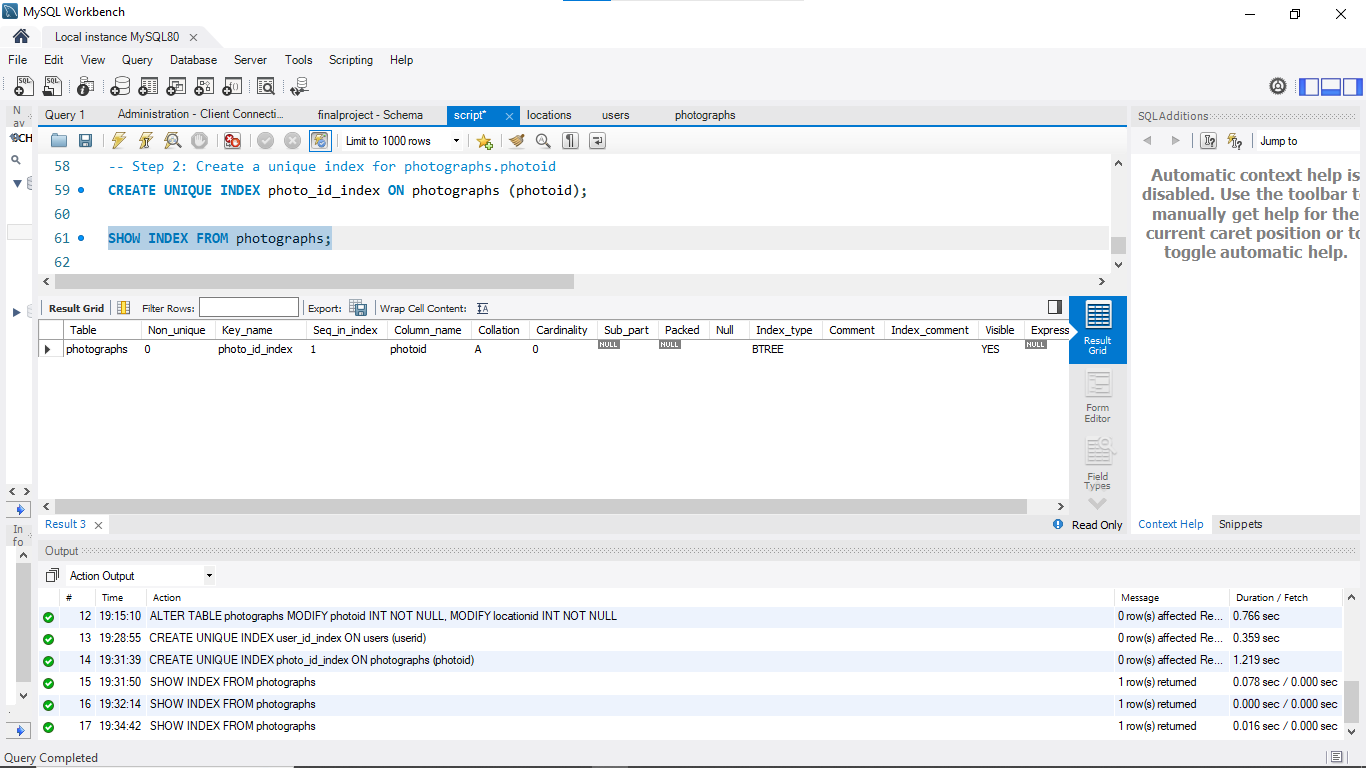
CREATE UNIQUE INDEX user\_id\_index ON users (userid);

SHOW INDEX FROM USERS;****

-- Step 2: Create a unique index for photographs.photoid

CREATE UNIQUE INDEX photo\_id\_index ON photographs (photoid);

SHOW INDEX FROM photographs;



**Prompt 5 - Enter Data**

-- SINCE USER ID IS IN INCREAMENT WE DO NOT NEED TO ADD IT

-- Insert user 1

INSERT INTO users (name, username, address, city, state, zip)

VALUES ('Bonnie Buntcake', 'bbunt', '6709 Wonder Street', 'Wonderbread', 'OH', '46106');

**-- Insert user 2**

INSERT INTO users (name, username, address, city, state, zip)

VALUES ('Sam Smarf', 'ssmarf', '356 A Street', 'Beefy', 'PA', '19943');

-- Insert user 3

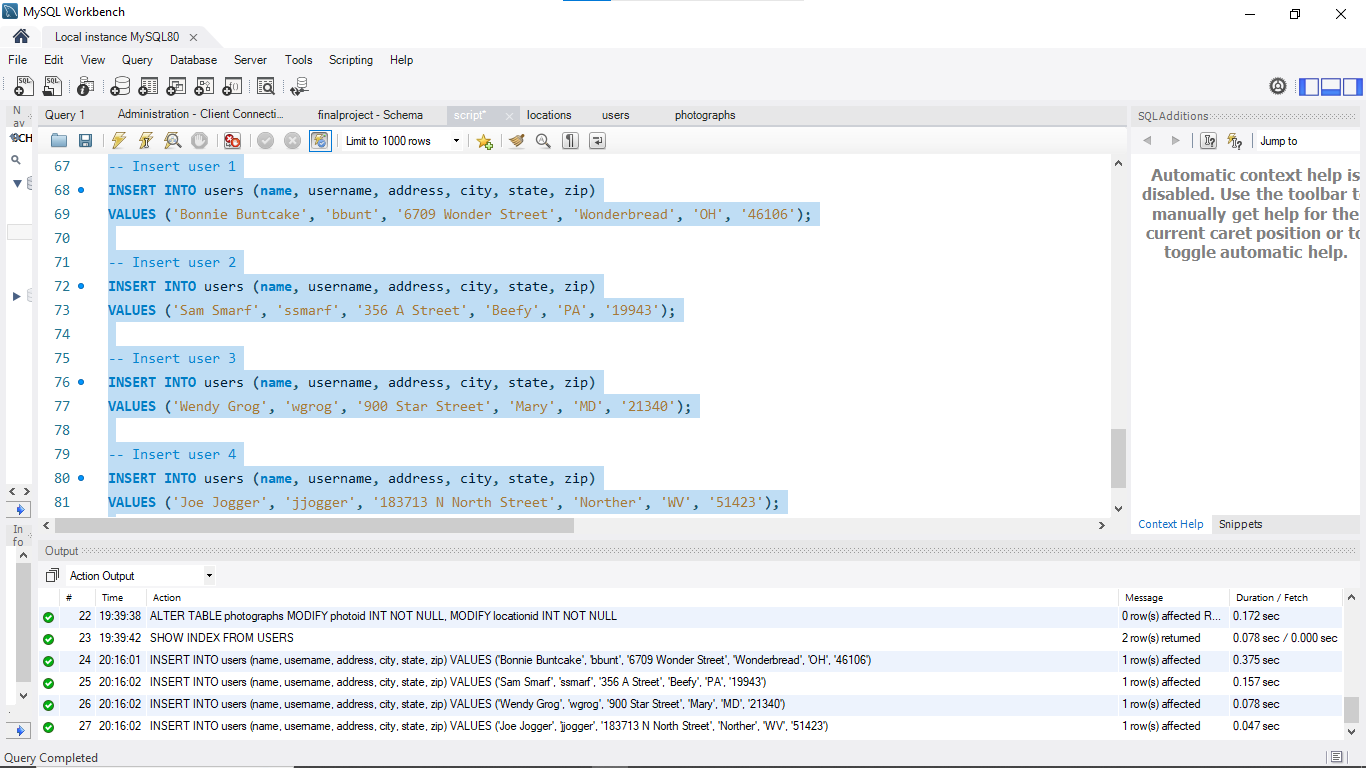
INSERT INTO users (name, username, address, city, state, zip)

VALUES ('Wendy Grog', 'wgrog', '900 Star Street', 'Mary', 'MD', '21340');

**-- Insert user 4**

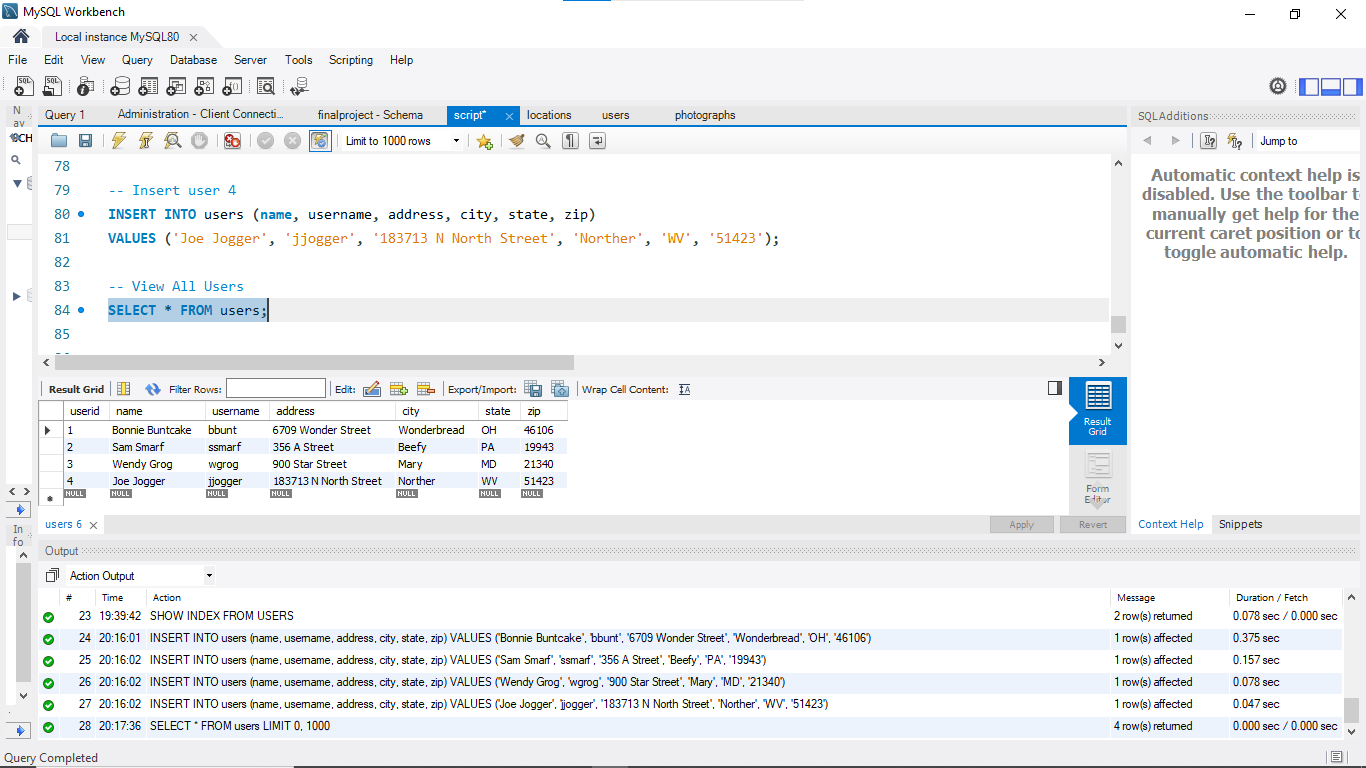
INSERT INTO users (name, username, address, city, state, zip)

VALUES ('Joe Jogger', 'jjogger', '183713 N North Street', 'Norther', 'WV', '51423');

****

**-- View All Users**

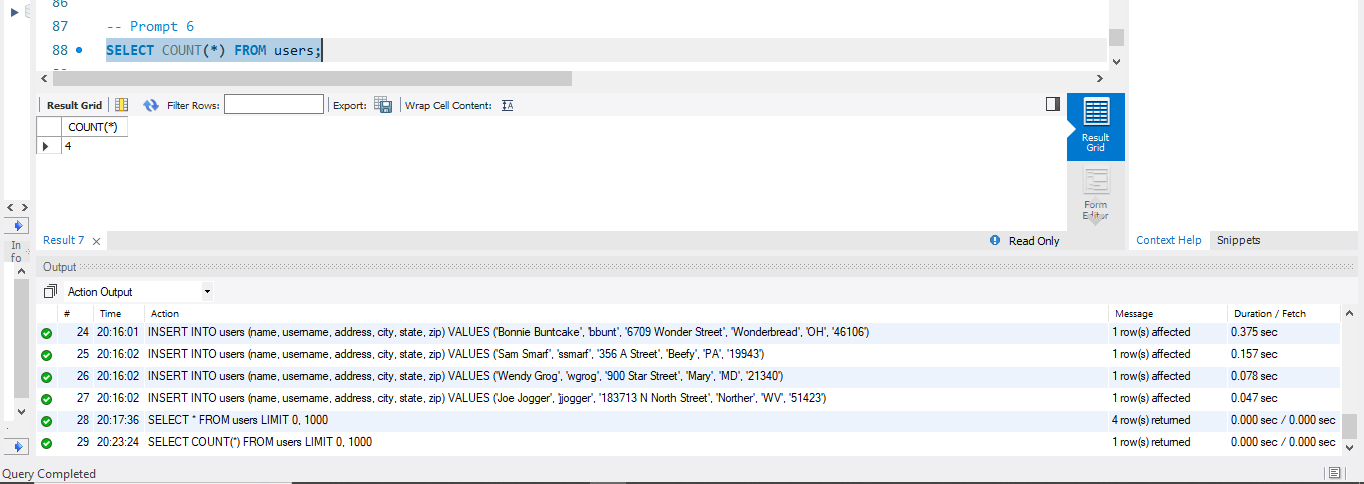
SELECT \* FROM users;

****

**Prompt 6: Count Rows**

-- Count the number of rows in the users table

SELECT COUNT(\*) FROM users;

****



**PROMPT 7:**

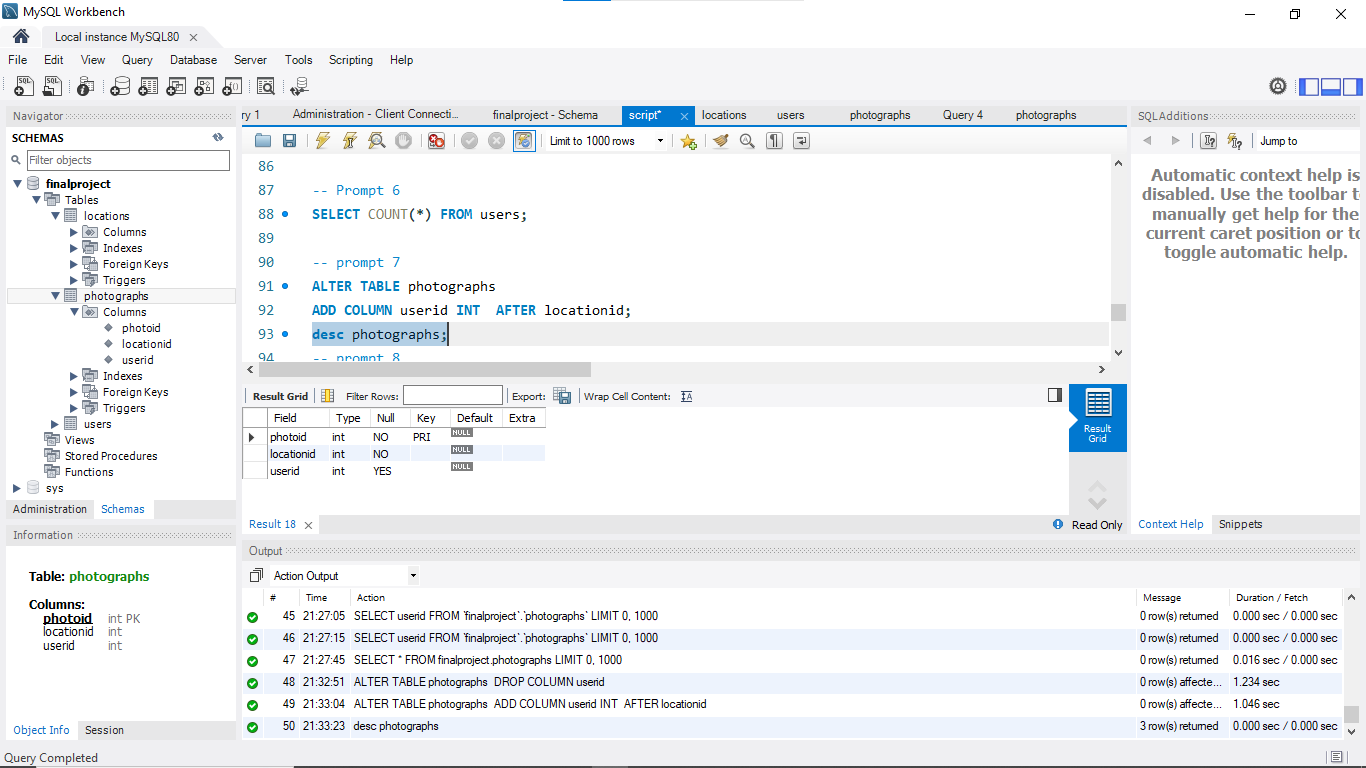
Add a new column called userid to the photographs table, so we can associate each photo with a user.

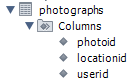
-- Add the `userid` column to the photographs table

ALTER TABLE photographs

ADD COLUMN userid INT AFTER locationid;

desc photographs;



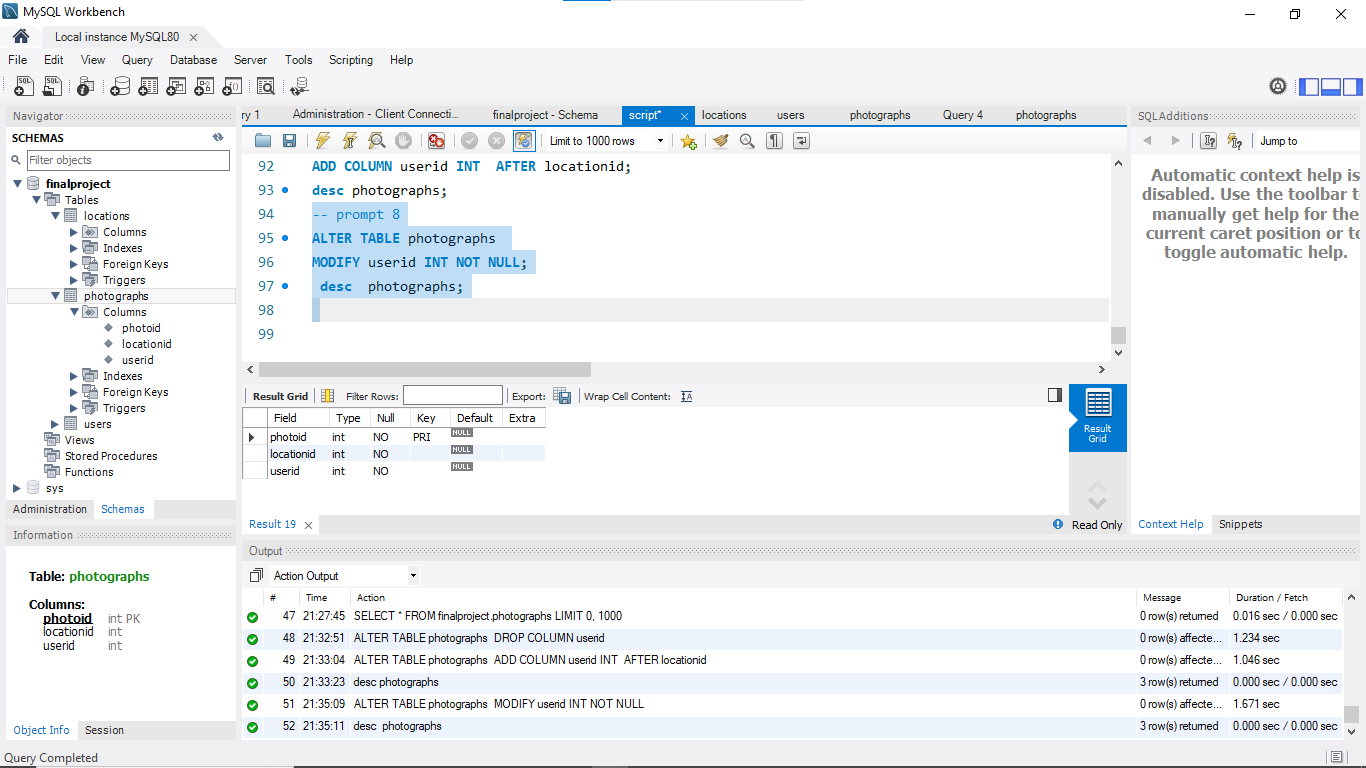


**PROMPT 8:**

ALTER TABLE photographs

MODIFY userid INT NOT NULL;

desc photographs;

****

In prompt seven, I did add a column “userid” to the table “photographs” without including the constraint ‘not null’. This could lead to a potential issue where by the photographs that could be inserted into the table without being associated with any user, would violate referential integrity. Thus, it would also weaken the relationship between users and their photos.

When we talk of a properly designed database, every photograph should be linked to a specific user who took it.

To solve this problem so as to ensure there is data integrity, I decided to use the ALTER TABLE ... MODIFY SQL command to update the column definition and explicitly prevent NULL values from being stored in the userid field.

**Prompt 9:**

-- Insert into locations and photographs

INSERT INTO locations (type, description, lng, lat)

VALUES

(1, 'Independence Hall', 794.35, 651.43),

(2, '6709 Wonder Street', 323.41, 412.22),

(1, 'Sunrise', 221.45, 132.43),

(2, '356 A Street', 123.32, 222.43),

(1, 'Mountains', 34.12, 87.99),

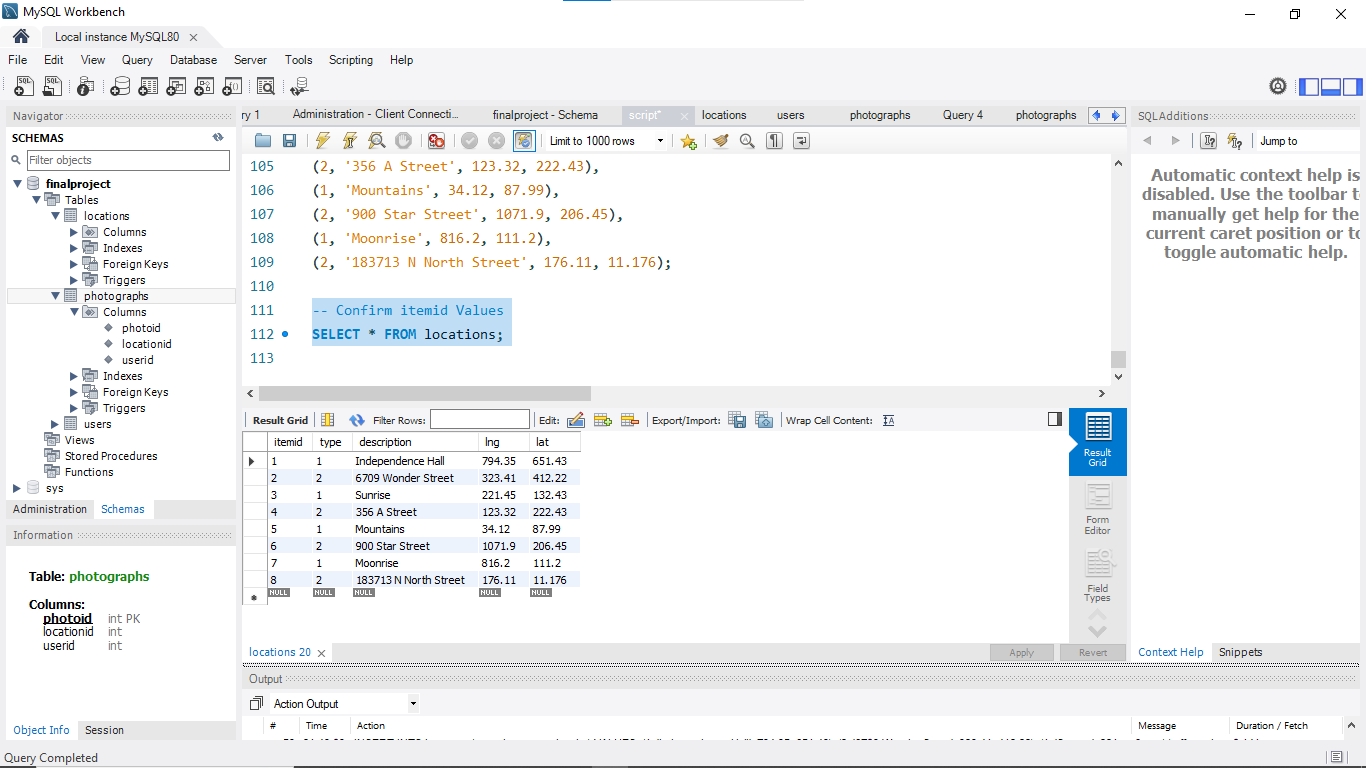
(2, '900 Star Street', 1071.9, 206.45),

(1, 'Moonrise', 816.2, 111.2),

(2, '183713 N North Street', 176.11, 11.176);

-- Confirm itemid Values

SELECT \* FROM locations;

****

**-- Insert into photographs**

INSERT INTO photographs (photoid, locationid, userid)

VALUES

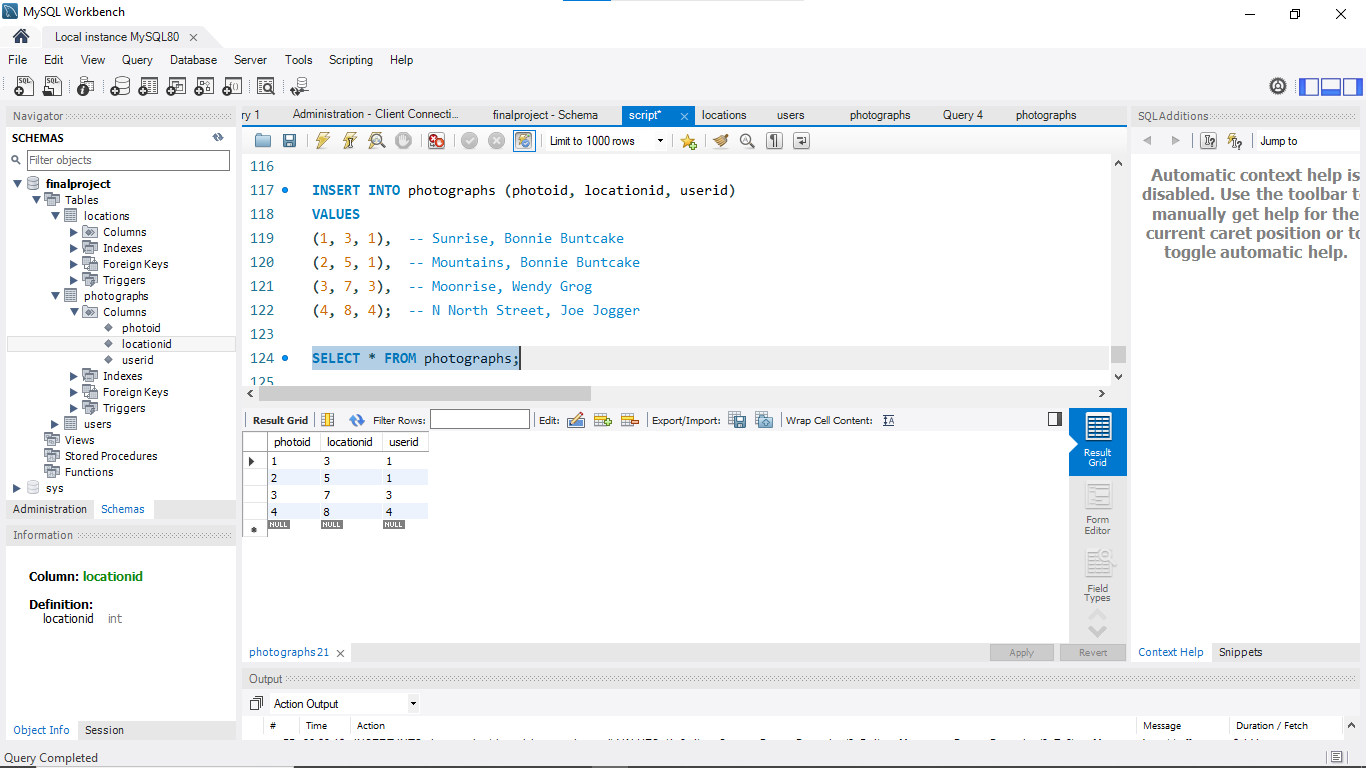
(1, 3, 1), -- Sunrise, Bonnie Buntcake

(2, 5, 1), -- Mountains, Bonnie Buntcake

(3, 7, 3), -- Moonrise, Wendy Grog

(4, 8, 4); -- N North Street, Joe Jogger

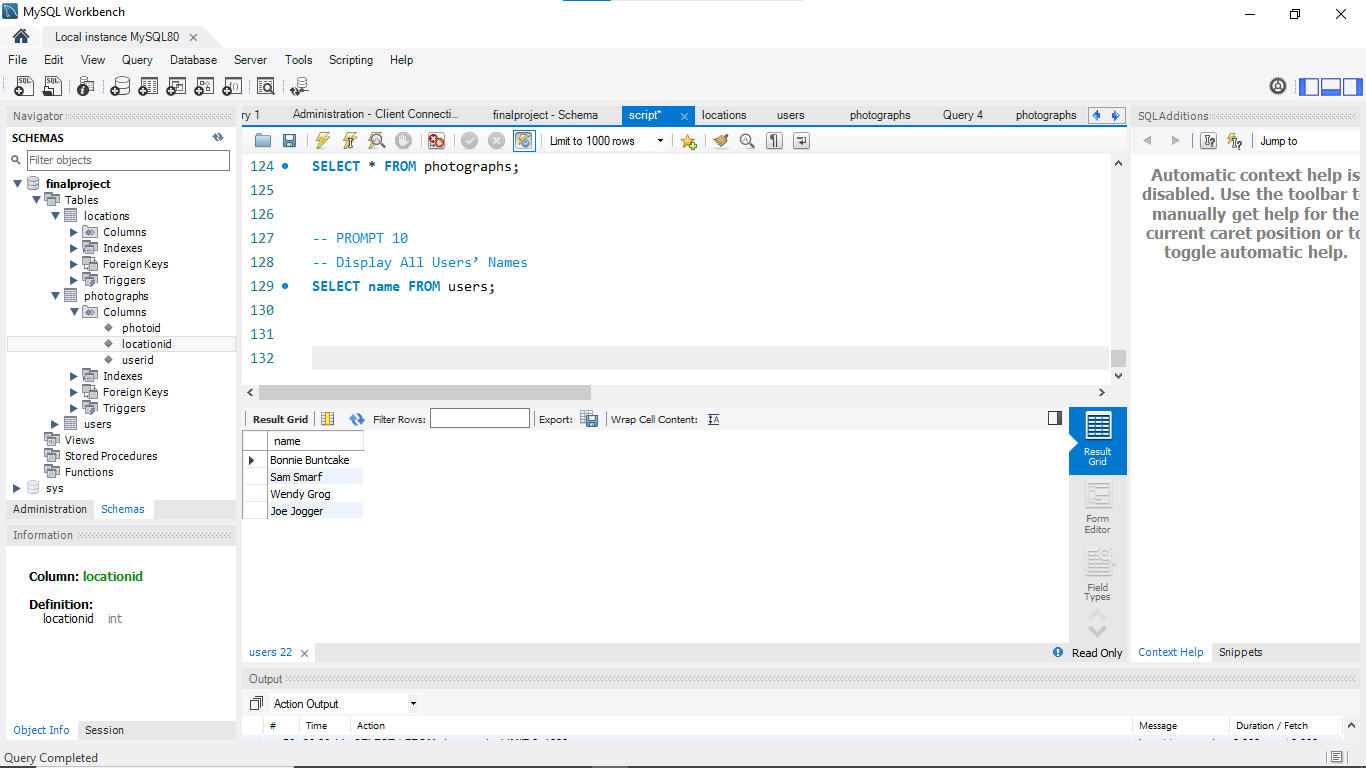
SELECT \* FROM photographs;



**-- PROMPT 10**

-- Display All Users’ Names

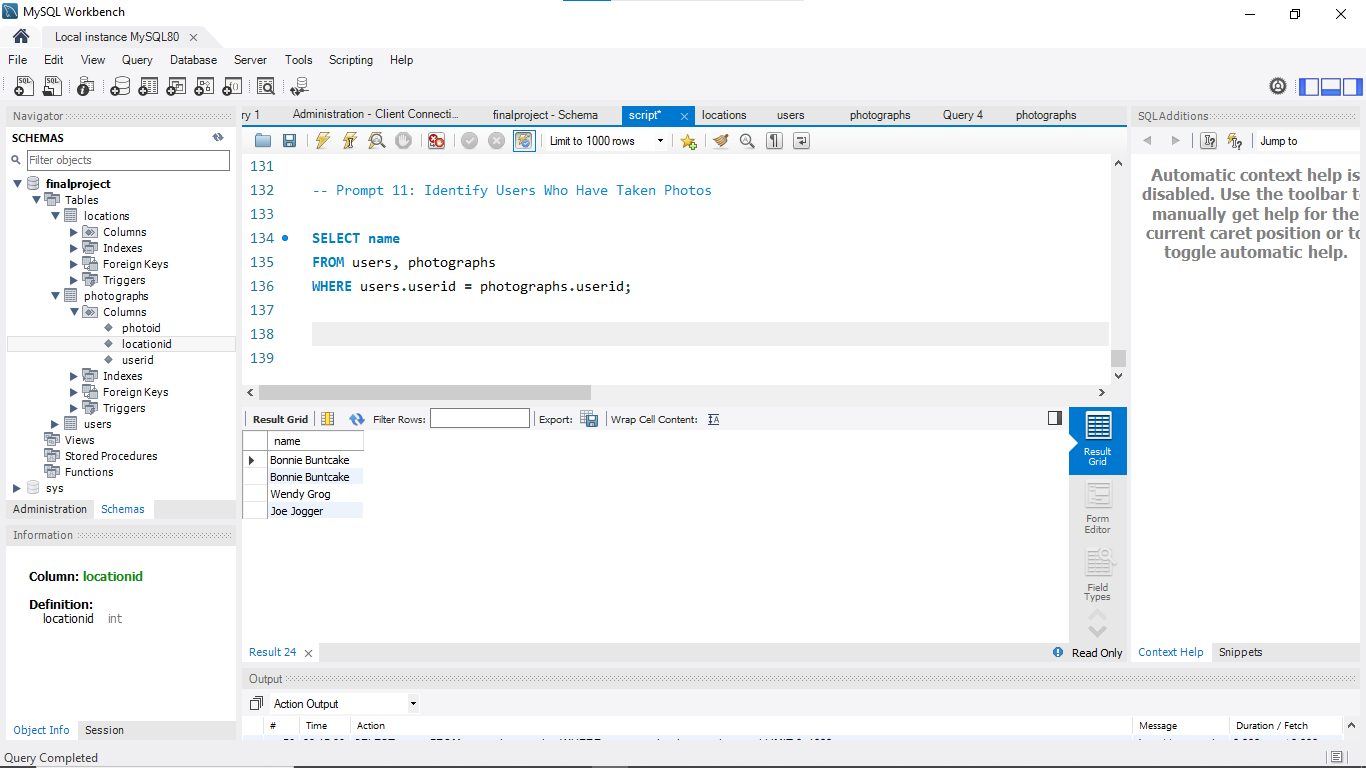
SELECT name FROM users;



**-- Prompt 11: Identify Users Who Have Taken Photos**

SELECT name

FROM users, photographs

WHERE users.userid = photographs.userid; 

**-- Prompt 12: Unique Names**

-- Query to get unique names of users who have taken pictures

SELECT DISTINCT name

FROM users, photographs

WHERE users.userid = photographs.userid;

