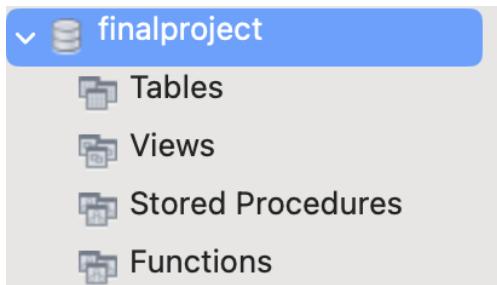


**MICHAEL DIAZ
COMPUTER SCIENCE 303 - ASSIGNMENT 1
JANUARY 15TH. 2023
STUDY.COM**

PROMPT 1 - Create the Schema/Database



PROMPT 2 - Create Tables

The screenshot shows the MySQL Workbench interface with the "Prompt1" tab selected. The left pane displays the "SCHEMAS" tree, where the "finalproject" schema is expanded to show "Tables", "Views", "Stored Procedures", and "Functions". The "Tables" item is currently selected. The right pane contains the SQL code for creating three tables:

```
1  CREATE TABLE users (
2      userID INT UNIQUE NOT NULL AUTO_INCREMENT,
3      name VARCHAR(50),
4      username CHAR(20),
5      address VARCHAR(255),
6      city VARCHAR(50),
7      state CHAR(2),
8      zip INT(5),
9
10     PRIMARY KEY (userID)
11 );
12
13 • CREATE TABLE locations (
14     itemID INT NOT NULL AUTO_INCREMENT,
15     type INT(10),
16     description VARCHAR(100),
17     lng REAL,
18     lat REAL,
19
20     PRIMARY KEY (itemID)
21 );
22
23
24 • CREATE TABLE photograph (
25     photoID INT(10),
26     locationID INT(10),
27
28     PRIMARY KEY (photoID)
29 );
```

PROMPT 3 - Alter Tables

```
ALTER TABLE locations
MODIFY type INT(10) NOT NULL,
MODIFY description VARCHAR(100) NOT NULL,
MODIFY lng REAL NOT NULL,
MODIFY lat REAL NOT NULL;
```

```
ALTER TABLE users
MODIFY name VARCHAR(50) NOT NULL,
MODIFY username CHAR(20) NOT NULL;
```

```
ALTER TABLE photograph
MODIFY photoID INT(10) NOT NULL,
MODIFY locationID INT(10) NOT NULL;
```

PROMPT 4 - Create Index

```
CREATE UNIQUE INDEX id ON users (userid);
CREATE UNIQUE INDEX photoTableID ON photograph (photoID);
```

CS303 Final - Warning - not supported

Administration Schemas Query 1 SQL File 9* users SQL File 10* SQL File 17* finalproject

SCHEMAS

Filter objects

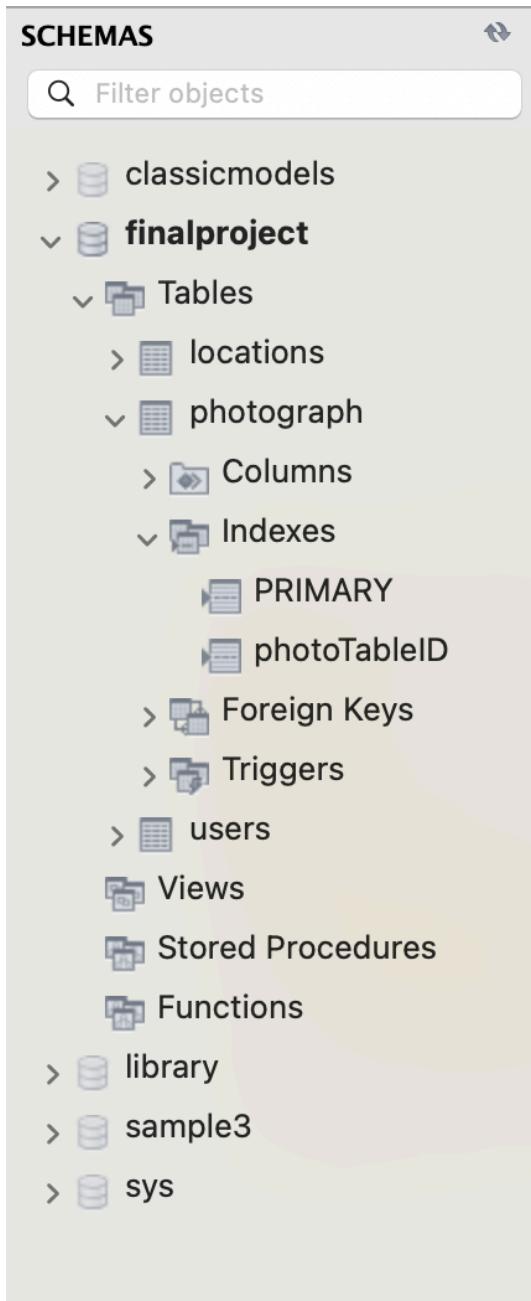
- Tables
 - locations
 - photograph
 - Columns
 - Indexes
 - PRIMARY
 - photoTableID
 - userID
 - Foreign Keys
 - Triggers
 - users
 - Columns
 - Indexes
 - PRIMARY
 - userID
 - id

Indexes in Table

Key	Type	Uniq...	Columns	Index Details
PRIMARY	BTREE	YES	photolD	Key Name: Index Type: Allows NULL: Cardinality: Comment: User Comment:
photoTableID	BTREE	YES	photolD	
userID	BTREE	YES	userID	

Columns in table

Column	Type	Nullable	Indexes
photolD	int	NO	PRIMARY, photoTableID
locationID	int	NO	
userID	int	NO	userID



PROMPT 5 - Enter Data

```
1 • INSERT INTO users (name, username, address, city, state, zip) VALUES ('Bonnie Buntcake', 'bbunt', '6709 Wonder Street', 'Wonderbread', 'Ohio', '46106');
2 • INSERT INTO users (name, username, address, city, state, zip) VALUES ('Sam Smarf', 'ssmarf', '356 A Street', 'Beefy', 'PA', '19943');
3 • INSERT INTO users (name, username, address, city, state, zip) VALUES ('Wendy Grog', 'wgrog', '900 Star Street', 'Mary', 'MD', '21340');
4 • INSERT INTO users (name, username, address, city, state, zip) VALUES ('Joe Jogger', 'jjogger', '183713 N North Street', 'Norther', 'WV', '51423');
```

```
1 • SELECT * FROM users;
```

userID	name	username	address	city	state	zip
1	Bonnie Buntcake	bbunt	6709 Wonder Street	Wonderbread	OH	46106
2	Sam Smarf	ssmarf	356 A Street	Beefy	PA	19943
3	Wendy Grog	wgrog	900 Star Street	Mary	MD	21340
4	Joe Jogger	jjogger	183713 N North Street	Norther	WV	51423
NULL	NULL	NULL	NULL	NULL	NULL	NULL

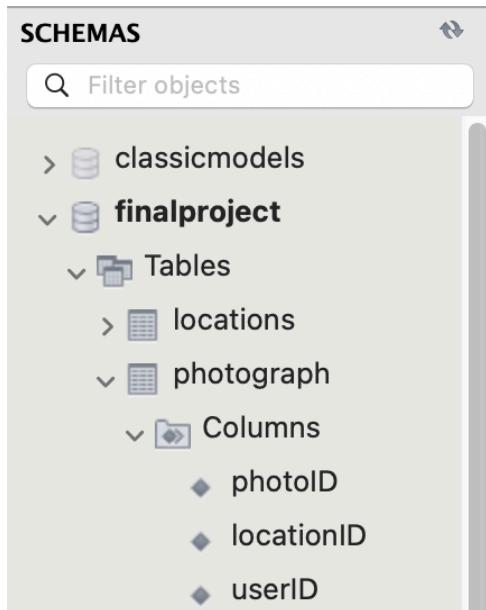
PROMPT 6 - Count Rows

```
1 • SELECT COUNT(*) FROM users;
```

COUNT(*)
4

PROMPT 7 - Add Column

```
ALTER TABLE photograph ADD COLUMN userID VARCHAR(20) AFTER locationID;
```



PROMPT 8 - Issue with New Column

The locations table and the photograph tables need to have a relationship in order to ensure we can find the location of where the photo was shot. We first need to alter the primary key of the locations table to an ID which suits the table better than 'itemID.' Since the photograph table already has a field named locationID, it would be best to change 'itemID' in the locations table to 'locationID,' then we should create a relationship between both tables by making locationID a primary key in the locations table, and a foreign key in the photograph table.

Also, the userID fields in the photograph table and users table do not have a relationship. The userID field in the photograph table needs to be a foreign key in the photograph table, referencing the primary key 'userID' from the users table. To do so, we will need to alter the table and ensure the data types between 'userID' in both tables are the same to allow the relationship. Next we need to alter the photograph table to create the relationship by making the userID column in the photograph table a foreign key, referencing the userID primary key in the users table.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the schema tree for 'finalproject'. The 'Tables' section contains 'locations' and 'photograph'. The 'Columns' section for 'locations' includes 'locationID' (selected), 'type', 'description', 'lng', and 'lat'. The 'Photograph' table has columns 'itemID', 'locationID', 'photographerID', 'date', and 'rating'. The 'Users' table has columns 'userID', 'username', 'password', 'email', and 'locationID'. The 'Views' and 'Stored Procedures' sections are empty.

```

1 #alters photograph table. Changes userID data type to match user ID data type in users table
2 • ALTER TABLE photograph
3 MODIFY COLUMN user ID INT UNIQUE AUTO_INCREMENT;
4
5 #adds the user ID field as a foreign key in the photograph table to create a relationship with the users table
6 • ALTER TABLE photograph
7 ADD FOREIGN KEY (user ID)
8 REFERENCES users (user ID);
9
10 #changes the field in the locations table from itemID to locationID, keeping the same primary key constraint, and data type
11 • ALTER TABLE locations
12 CHANGE COLUMN itemID locationID INT NOT NULL AUTO_INCREMENT;
13
14 #adds the locationID field as a foreign key in the photograph table to create a relationship with the locations table table
15 • ALTER TABLE photograph
16 ADD FOREIGN KEY (locationID)
17 REFERENCES locations (locationID);

```

PROMPT 9 - Location and Photograph Table Updates

```
# inserts data into locations table
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', '183714 N North Street', '176.11', '11.176');
```

Data in locations table after insert:

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' tree shows the 'finalproject' schema selected. In the main pane, a SQL editor window displays the following code:

```

1
2 •  SELECT * FROM locations;
3
4

```

Below the SQL editor is a 'Result Grid' showing the data from the 'locations' table:

locationID	type	description	lng	lat
1	1	Independence Hall	794.35	651.43
2	2	6709 Wonder Street	329.41	412.22
3	1	Sunrise	221.45	132.43
4	2	356 A Street	123.32	222.43
5	1	Mountaine	34.12	87.99
6	2	900 Star Street	1071.9	206.45
7	1	Moonrise	816.2	111.2
8	2	183714 N North Street	176.11	11.176

At the bottom, the 'Action Output' section shows the execution details:

Time	Action	Response	Duration / Fetch Time
18:59:37	SELECT * FROM locations LIMIT 0, 1000	8 row(s) returned	0.0013 sec / 0.00001...

inserts data into photograph table

```

INSERT INTO photograph (photoID, locationID, userID) VALUES (1, 3, 1);
INSERT INTO photograph (photoID, locationID, userID) VALUES (2, 6, 1);
INSERT INTO photograph (photoID, locationID, userID) VALUES (3, 7, 3);
INSERT INTO photograph (photoID, locationID, userID) VALUES (4, 4, 4);

```

Data in photograph table after insert:

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' tree shows the 'finalproject' schema selected. In the main pane, a SQL editor window displays the following code:

```

1
2 •  SELECT * FROM photograph;
3
4

```

Below the SQL editor is a 'Result Grid' showing the data from the 'photograph' table:

photoID	locationID	userID
1	3	1
2	6	1
3	7	3
4	4	4

At the bottom, the 'Action Output' section shows the execution details:

Time	Action	Response	Duration / Fetch Time
18:59:37	SELECT * FROM locations LIMIT 0, 1000	8 row(s) returned	0.0013 sec / 0.00001...
19:00:14	SELECT * FROM photograph LIMIT 0, 1000	4 row(s) returned	0.00087 sec / 0.0000...

PROMPT 10 - Users

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' tree view is open, showing the 'finalproject' schema expanded, revealing tables like 'locations', 'photograph', and 'users'. The main area displays a SQL editor with the following query:

```
1
2 •  SELECT name FROM users;
3
4
```

Below the SQL editor is a 'Result Grid' showing the names of the users:

name
Bonnie Buntcake
Sam Smarf
Wendy Grog
Joe Jogger

PROMPT 11 - Who's taking pictures?

The screenshot shows the MySQL Workbench interface with a more complex schema. The 'SCHEMAS' tree view shows the 'finalproject' schema expanded, including the 'photograph' table with its columns, indexes, and foreign keys (like 'photograph_ibfk_1'). The main SQL editor contains the following query:

```
1 •  SELECT name
2   FROM users, photograph
3  WHERE users.userID = photograph.userID;
```

The 'Result Grid' below shows the names of the users who have taken pictures:

name
Bonnie Buntcake
Bonnie Buntcake
Wendy Grog
Joe Jogger

PROMPT 12 - Unique Names

CS303 Final - Warning - not supported

Administration Schemas

SCHEMAS

Filter objects

> classicmodels

< finalproject

< Tables

> locations

< photograph

< Columns

< Indexes

< Foreign Keys

< photograph_ibfk_1

< Triggers

< users

< Views

< Stored Procedures

< Functions

< library

< sample3

Object Info Session

No object selected

Query 1 SQL File 9* Prompt9* SQL File 22* SQL File 6* SQL File 8* photograph SQL File 10* SQL File 11* >

1 • **SELECT DISTINCT name**

2 **FROM users, photograph**

3 **WHERE users.userID = photograph.userID;**

100% 40:3

Result Grid Filter Rows: Search Export:

name

Bonnie Buntcake
Wendy Grog
Joe Jogger

Result 11 Read Only

Action Output

Time	Action	Response	Duration / Fetch Time
19:33:20	SELECT DISTINCT name FROM users, photograph WHERE users.userID =...	3 row(s) returned	0.0013 sec / 0.00000...

Result Grid Form Editor Find Tables