

**Art Gallery Management System**

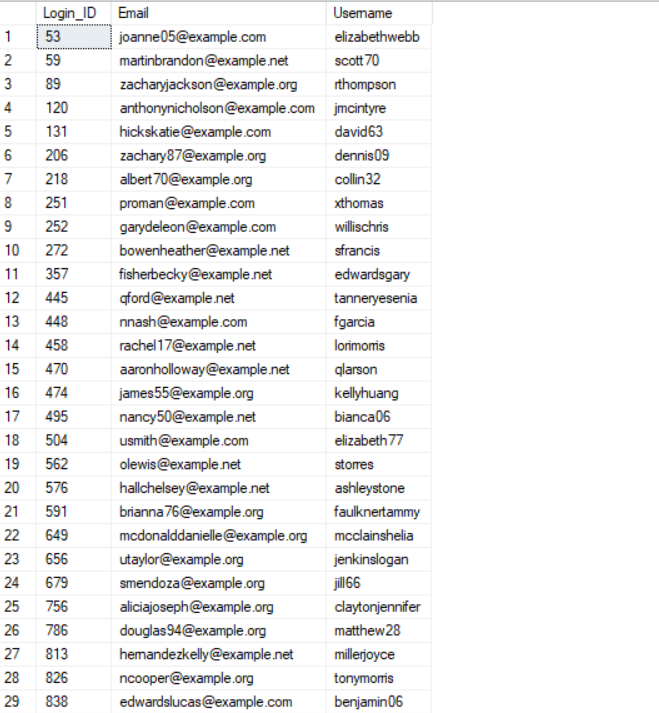
The goal with our Art Gallery Management System is to develop a management system that catalogs artists, their works, information about the art gallery, exhibitions, and displays paintings' images for patron viewing.

**Objective:** The primary goal is to inform art enthusiasts about various exhibition details, including the artwork and artists, that are arranged by various organizations. Our system will make it easy for customers to get in touch with the artist to purchase their artwork. For administrators, artists, and customers, there is a login page.

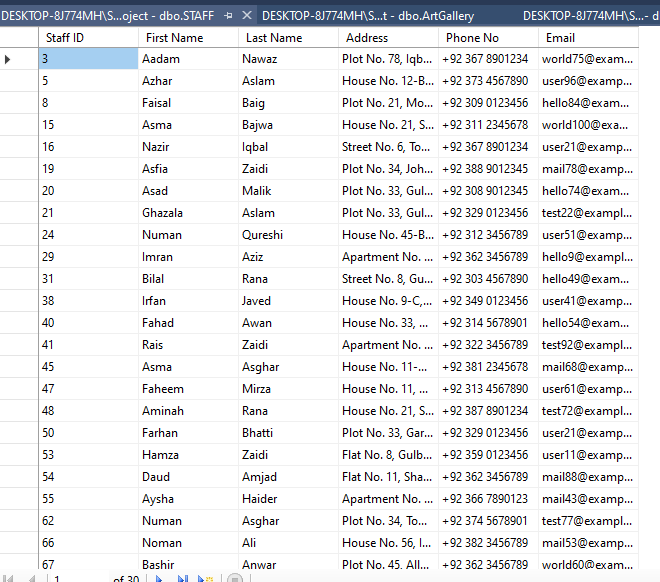
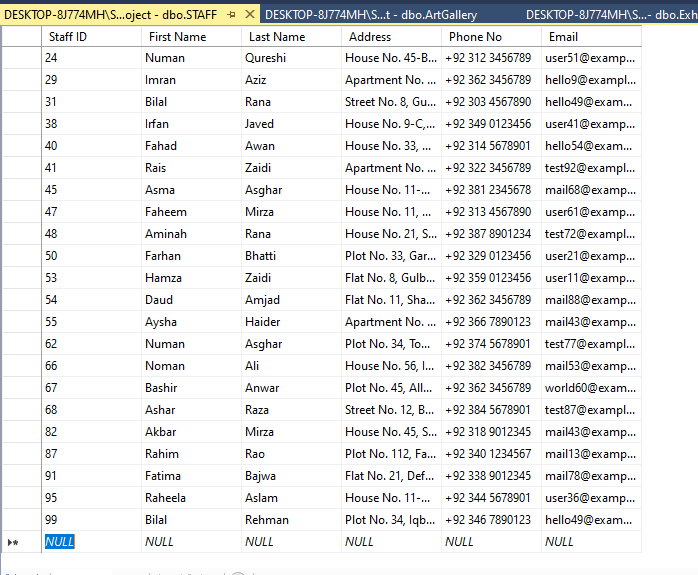
**Normalized ER Diagram:**



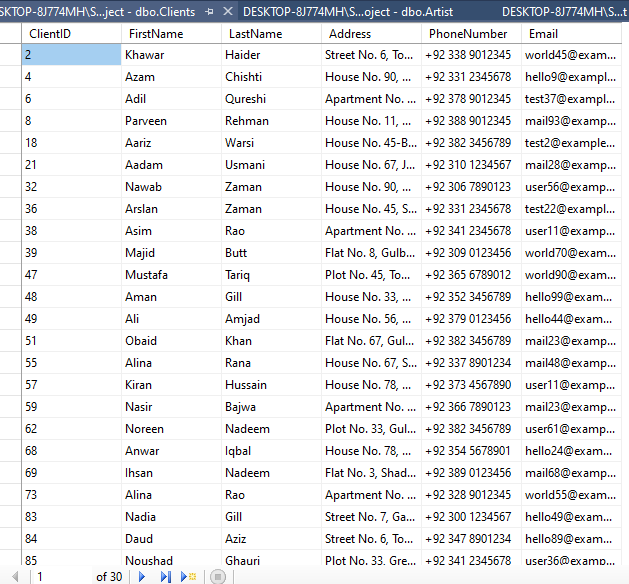
**Python Connection to Insert Data:**

**Login Table:  
**

import random  
import pyodbc as odbc  
from faker import Faker  
  
fake = Faker()  
  
def generate\_login\_record(used\_ids):  
 while True:  
 random\_login\_id = random.randint(1, 1000)  
 if random\_login\_id not in used\_ids:  
 break  
 used\_ids.add(random\_login\_id)  
 random\_email = fake.email()  
 random\_username = fake.user\_name()  
 return {"Login\_ID": random\_login\_id, "Email": random\_email, "Username": random\_username}  
  
# Generate 30 login records  
used\_ids = set()  
login\_records = [generate\_login\_record(used\_ids) for \_ in range(30)]  
  
# Database connection details  
SERVER\_NAME = r'DESKTOP-L5NG8PP\SQLEXPRESS'  
DATABASE\_NAME = 'ArtGallery'  
  
# Connection string  
conn\_str = (  
 f'DRIVER={{ODBC Driver 17 for SQL Server}};'  
 f'SERVER={SERVER\_NAME};'  
 f'DATABASE={DATABASE\_NAME};'  
 r'Trusted\_Connection=yes;'  
)  
  
# Establish connection  
conn = odbc.connect(conn\_str)  
cursor = conn.cursor()  
  
# Create Login table if it doesn't exist  
cursor.execute('''  
 IF NOT EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Login')  
 CREATE TABLE Login  
 (Login\_ID INT PRIMARY KEY,  
 Email VARCHAR(255) UNIQUE,  
 Username VARCHAR(100))  
 ''')  
  
# Insert records into the Login table  
for record in login\_records:  
 cursor.execute('''  
 INSERT INTO Login (Login\_ID, Email, Username)  
 VALUES (?, ?, ?)  
 ''',  
 record["Login\_ID"], record["Email"], record["Username"])  
 print("Inserted Successfully")  
  
# Commit the transaction  
conn.commit()  
  
# Close the cursor and connection  
cursor.close()  
conn.close()

**Staff Table:  
**

import random  
import pyodbc as odbc  
  
#List of first\_name  
S\_firstname = ["Aadam", "Aalia", "Aamir", "Aariz", "Aasia", "Abdul", "Abeeha", "Adeel", "Adil", "Afreen",  
 "Afzal", "Ahmed", "Aisha", "Akbar", "Aleeza", "Ali", "Alina", "Alisha", "Aman", "Amina",  
 "Aminah", "Amir", "Ammar", "Amna", "Anaya", "Ansa", "Anwar", "Aqsa", "Arham", "Arif",  
 "Arisha", "Arslan", "Asad", "Asfia", "Ashar", "Ashraf", "Asim", "Asma", "Asrar", "Atif",  
 "Ayesha", "Ayman", "Aysha", "Azam", "Azhar", "Aziz", "Baqir", "Bashir", "Bilal", "Danish",  
 "Daud", "Ehsan", "Faheem", "Fahad", "Faiza", "Faisal", "Faizan", "Farah", "Farhan", "Faris",  
 "Fatima", "Fawad", "Fiza", "Ghazala", "Gulzar", "Habib", "Hadi", "Hafsa", "Hammad", "Hamza",  
 "Haniya", "Haris", "Hassan", "Hiba", "Hira", "Humaira", "Hussain", "Huzefa", "Ibad", "Ibrahim",  
 "Iftikhar", "Ihsan", "Imran", "Inam", "Inayat", "Iqbal", "Irfan", "Ismail", "Jahanzeb",  
 "Jahangir", "Jamal", "Jamil", "Jawad", "Junaid", "Kamran", "Kashif", "Kausar", "Khalid",  
 "Khawar", "Khurram", "Kiran", "Komal", "Lubna", "Mahmood", "Mahnoor", "Majid", "Makbool",  
 "Malik", "Manzar", "Marium", "Mateen", "Mehboob", "Mehreen", "Mian", "Minhaj", "Moin", "Moiz",  
 "Munawar", "Musa", "Mushtaq", "Mustafa", "Nadia", "Naeem", "Nagina", "Naima", "Najam", "Naseer",  
 "Nasir", "Nawab", "Nawaz", "Nazia", "Nazir", "Nida", "Nisar", "Noman", "Noreen", "Noushad",  
 "Numan", "Nusrat", "Obaid", "Omer", "Owais", "Parveen", "Qaisar", "Qamar", "Qasim", "Raheela",  
 "Raheem", "Rahim", "Rais"]  
  
#List of lastnames  
S\_lastname = ["Abbasi", "Afzal", "Ahmad", "Ali", "Amjad", "Anwar", "Asghar", "Aslam", "Awan", "Aziz",  
 "Baig", "Bajwa", "Bhatti", "Butt", "Chaudhry", "Chishti", "Farooqi", "Ghauri", "Gill", "Haider",  
 "Hussain", "Iqbal", "Javed", "Khan", "Khawaja", "Malik", "Mirza", "Nadeem", "Nawaz", "Qureshi",  
 "Rana", "Rao", "Raza", "Rehman", "Saeed", "Saleem", "Siddiqui", "Tariq", "Usmani", "Warsi",  
 "Yousaf", "Zafar", "Zaidi", "Zaman", "Abbasi", "Afzal", "Ahmad", "Ali", "Amjad", "Anwar",  
 "Asghar", "Aslam", "Awan", "Aziz", "Baig", "Bajwa", "Bhatti", "Butt", "Chaudhry", "Chishti",  
 "Farooqi", "Ghauri", "Gill", "Haider", "Hussain", "Iqbal", "Javed", "Khan", "Khawaja", "Malik",  
 "Mirza", "Nadeem", "Nawaz", "Qureshi", "Rana", "Rao", "Raza", "Rehman", "Saeed", "Saleem",  
 "Siddiqui", "Tariq", "Usmani", "Warsi", "Yousaf", "Zafar", "Zaidi", "Zaman"]  
  
#List of Staff Address  
S\_address = ["House No. 123, Block A, Model Town",  
 "Flat No. 56, Gulberg III",  
 "Street No. 7, Garden Town",  
 "Plot No. 89, Johar Town",  
 "Apartment No. 34, DHA Phase 6",  
 "House No. 12-B, Gulshan-e-Ravi",  
 "Plot No. 45, Township",  
 "House No. 78, Wapda Town",  
 "Street No. 23, Allama Iqbal Town",  
 "Flat No. 3, Shadman Colony",  
 "House No. 67, Sabzazar Scheme",  
 "Apartment No. 21, Gulshan-e-Lahore",  
 "House No. 9-C, Green Town",  
 "Plot No. 112, Faisal Town",  
 "Street No. 5, Samanabad",  
 "House No. 33, Garden Town",  
 "Flat No. 8, Gulberg II",  
 "Street No. 12, Iqbal Town",  
 "House No. 56, Model Town Extension",  
 "Apartment No. 7, DHA Phase 5",  
 "Plot No. 23, Johar Town Extension",  
 "Street No. 14, Wapda Town",  
 "House No. 90, Allama Iqbal Town",  
 "Flat No. 11, Shadman Colony",  
 "Plot No. 34, Township Extension",  
 "House No. 45-B, Nishtar Colony",  
 "Apartment No. 9, Gulshan-e-Ravi",  
 "House No. 21, Shahdara Town",  
 "Plot No. 7, Model Town",  
 "Street No. 8, Gulberg IV",  
 "Flat No. 4, Johar Town Phase 2",  
 "House No. 78, Defence View Housing Society",  
 "Apartment No. 32, Valencia Housing Society",  
 "House No. 56, Iqbal Avenue",  
 "Plot No. 12, Bahria Town",  
 "Street No. 21, Garden Town",  
 "Flat No. 67, Gulshan-e-Iqbal",  
 "House No. 11-A, Wapda Town Extension",  
 "Plot No. 33, Green Town",  
 "House No. 45, Samanabad Extension",  
 "Apartment No. 8, DHA Phase 1",  
 "House No. 22, Johar Town",  
 "Plot No. 78, Gulberg I",  
 "Street No. 6, Township",  
 "Flat No. 3, Nishtar Colony",  
 "House No. 89, Iqbal Town Extension",  
 "Apartment No. 11, Shadman Colony",  
 "House No. 34, Allama Iqbal Town",  
 "Plot No. 21, Model Town Extension",  
 "Street No. 9, Gulberg III",  
 "Flat No. 22, Garden Town",  
 "House No. 67, Johar Town Phase 1",  
 "Plot No. 45, Wapda Town Extension",  
 "Apartment No. 12, Defence View Housing Society",  
 "House No. 56, Valencia Housing Society",  
 "Plot No. 34, Iqbal Avenue",  
 "Street No. 12, Bahria Town",  
 "Flat No. 21, Garden Town",  
 "House No. 78, Gulshan-e-Iqbal",  
 "Apartment No. 56, Wapda Town Extension",  
 "House No. 11, Model Town",  
 "Plot No. 33, Gulberg IV",  
 "Street No. 8, Johar Town Phase 2",  
 "Flat No. 4, Defence View Housing Society",  
 "House No. 22, Valencia Housing Society",  
 "Plot No. 78, Iqbal Avenue",  
 "Street No. 6, Bahria Town",  
 "Apartment No. 3, Garden Town",  
 "House No. 89, Gulberg I",  
 "Plot No. 21, Township",  
 "Street No. 9, Nishtar Colony",  
 "Flat No. 22, Iqbal Town Extension",  
 "House No. 67, Shadman Colony",  
 "Plot No. 45, Allama Iqbal Town",  
 "Apartment No. 12, Model Town Extension",  
 "House No. 56, Gulberg III",  
 "Plot No. 34, Johar Town",  
 "Street No. 12, Wapda Town Extension",  
 "Flat No. 21, Defence View Housing Society",  
 "House No. 78, Valencia Housing Society",  
 "Apartment No. 56, Iqbal Avenue",  
 "House No. 11, Bahria Town",  
 "Plot No. 33, Garden Town",  
 "Street No. 8, Gulberg IV",  
 "Flat No. 4, Johar Town Phase 2"]  
  
#List of Phone Numbers  
S\_phoneno = ["+92 300 1234567",  
 "+92 301 2345678",  
 "+92 302 3456789",  
 "+92 303 4567890",  
 "+92 304 5678901",  
 "+92 305 6789012",  
 "+92 306 7890123",  
 "+92 307 8901234",  
 "+92 308 9012345",  
 "+92 309 0123456",  
 "+92 310 1234567",  
 "+92 311 2345678",  
 "+92 312 3456789",  
 "+92 313 4567890",  
 "+92 314 5678901",  
 "+92 315 6789012",  
 "+92 316 7890123",  
 "+92 317 8901234",  
 "+92 318 9012345",  
 "+92 319 0123456",  
 "+92 320 1234567",  
 "+92 321 2345678",  
 "+92 322 3456789",  
 "+92 323 4567890",  
 "+92 324 5678901",  
 "+92 325 6789012",  
 "+92 326 7890123",  
 "+92 327 8901234",  
 "+92 328 9012345",  
 "+92 329 0123456",  
 "+92 330 1234567",  
 "+92 331 2345678",  
 "+92 332 3456789",  
 "+92 333 4567890",  
 "+92 334 5678901",  
 "+92 335 6789012",  
 "+92 336 7890123",  
 "+92 337 8901234",  
 "+92 338 9012345",  
 "+92 339 0123456",  
 "+92 340 1234567",  
 "+92 341 2345678",  
 "+92 342 3456789",  
 "+92 343 4567890",  
 "+92 344 5678901",  
 "+92 345 6789012",  
 "+92 346 7890123",  
 "+92 347 8901234",  
 "+92 348 9012345",  
 "+92 349 0123456",  
 "+92 350 1234567",  
 "+92 351 2345678",  
 "+92 352 3456789",  
 "+92 353 4567890",  
 "+92 354 5678901",  
 "+92 355 6789012",  
 "+92 356 7890123",  
 "+92 357 8901234",  
 "+92 358 9012345",  
 "+92 359 0123456",  
 "+92 360 1234567",  
 "+92 361 2345678",  
 "+92 362 3456789",  
 "+92 363 4567890",  
 "+92 364 5678901",  
 "+92 365 6789012",  
 "+92 366 7890123",  
 "+92 367 8901234",  
 "+92 368 9012345",  
 "+92 369 0123456",  
 "+92 370 1234567",  
 "+92 371 2345678",  
 "+92 372 3456789",  
 "+92 373 4567890",  
 "+92 374 5678901",  
 "+92 375 6789012",  
 "+92 376 7890123",  
 "+92 377 8901234",  
 "+92 378 9012345",  
 "+92 379 0123456",  
 "+92 380 1234567",  
 "+92 381 2345678",  
 "+92 382 3456789",  
 "+92 383 4567890",  
 "+92 384 5678901",  
 "+92 385 6789012",  
 "+92 386 7890123",  
 "+92 387 8901234",  
 "+92 388 9012345",  
 "+92 389 0123456"  
 ]  
  
#List of EmailID  
S\_email = ["user1@example.com",  
 "test2@example.com",  
 "mail3@example.com",  
 "hello4@example.com",  
 "world5@example.com",  
 "user6@example.com",  
 "test7@example.com",  
 "mail8@example.com",  
 "hello9@example.com",  
 "world10@example.com",  
 "user11@example.com",  
 "test12@example.com",  
 "mail13@example.com",  
 "hello14@example.com",  
 "world15@example.com",  
 "user16@example.com",  
 "test17@example.com",  
 "mail18@example.com",  
 "hello19@example.com",  
 "world20@example.com",  
 "user21@example.com",  
 "test22@example.com",  
 "mail23@example.com",  
 "hello24@example.com",  
 "world25@example.com",  
 "user26@example.com",  
 "test27@example.com",  
 "mail28@example.com",  
 "hello29@example.com",  
 "world30@example.com",  
 "user31@example.com",  
 "test32@example.com",  
 "mail33@example.com",  
 "hello34@example.com",  
 "world35@example.com",  
 "user36@example.com",  
 "test37@example.com",  
 "mail38@example.com",  
 "hello39@example.com",  
 "world40@example.com",  
 "user41@example.com",  
 "test42@example.com",  
 "mail43@example.com",  
 "hello44@example.com",  
 "world45@example.com",  
 "user46@example.com",  
 "test47@example.com",  
 "mail48@example.com",  
 "hello49@example.com",  
 "world50@example.com",  
 "user51@example.com",  
 "test52@example.com",  
 "mail53@example.com",  
 "hello54@example.com",  
 "world55@example.com",  
 "user56@example.com",  
 "test57@example.com",  
 "mail58@example.com",  
 "hello59@example.com",  
 "world60@example.com",  
 "user61@example.com",  
 "test62@example.com",  
 "mail63@example.com",  
 "hello64@example.com",  
 "world65@example.com",  
 "user66@example.com",  
 "test67@example.com",  
 "mail68@example.com",  
 "hello69@example.com",  
 "world70@example.com",  
 "user71@example.com",  
 "test72@example.com",  
 "mail73@example.com",  
 "hello74@example.com",  
 "world75@example.com",  
 "user76@example.com",  
 "test77@example.com",  
 "mail78@example.com",  
 "hello79@example.com",  
 "world80@example.com",  
 "user81@example.com",  
 "test82@example.com",  
 "mail83@example.com",  
 "hello84@example.com",  
 "world85@example.com",  
 "user86@example.com",  
 "test87@example.com",  
 "mail88@example.com",  
 "hello89@example.com",  
 "world90@example.com",  
 "user91@example.com",  
 "test92@example.com",  
 "mail93@example.com",  
 "hello94@example.com",  
 "world95@example.com",  
 "user96@example.com",  
 "test97@example.com",  
 "mail98@example.com",  
 "hello99@example.com",  
 "world100@example.com"  
 ]  
  
  
SERVER\_NAME = r'DESKTOP-8J774MH\SQLEXPRESS'  
DATABASE\_NAME = 'SemesterProject'  
  
# Define connection string  
conn\_str = (  
 f'DRIVER={{ODBC Driver 17 for SQL Server}};'  
 f'SERVER={SERVER\_NAME};'  
 f'DATABASE={DATABASE\_NAME};'  
 r'Trusted\_Connection=yes;' # For Windows Authentication  
)  
  
try:  
 # Establish connection  
 conn = odbc.connect(conn\_str)  
  
 # Create a cursor  
 cursor = conn.cursor()  
  
# Create STAFF table if it doesn't exist  
 cursor.execute("""  
 IF NOT EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'STAFF')  
 CREATE TABLE STAFF (  
 [Staff ID] INT PRIMARY KEY,  
 [First Name] NVARCHAR(50),  
 [Last Name] NVARCHAR(50),  
 [Address] NVARCHAR(100),  
 [Phone No] NVARCHAR(20),  
 [Email] NVARCHAR(100)  
 )  
 """)  
  
 # Commit the transaction  
 conn.commit()  
  
 # Keep track of used staff IDs  
 used\_staff\_ids = set()  
  
 def generate\_record():  
 while True:  
 random\_staffid = random.randint(0, 100)  
 if random\_staffid not in used\_staff\_ids:  
 used\_staff\_ids.add(random\_staffid)  
 break  
 random\_fname = random.choice(S\_firstname)  
 random\_lname = random.choice(S\_lastname)  
 random\_address = random.choice(S\_address)  
 random\_phoneno = random.choice(S\_phoneno)  
 random\_email = random.choice(S\_email)  
 return (random\_staffid, random\_fname, random\_lname, random\_address, random\_phoneno, random\_email)  
  
 # Generate 30 records and insert them into the database  
 for \_ in range(30):  
 record = generate\_record()  
 sql\_insert = """  
 INSERT INTO STAFF ([Staff ID], [First Name], [Last Name], [Address], [Phone No], [Email])  
 VALUES (?, ?, ?, ?, ?, ?)  
 """  
 cursor.execute(sql\_insert, record)  
 print("Record Inserted Successfully")  
  
 # Commit the transaction  
 conn.commit()  
  
except odbc.Error as e:  
 print("Error inserting to database:", e)  
  
finally:  
 # Close cursor and connection  
 cursor.close()  
 conn.close()

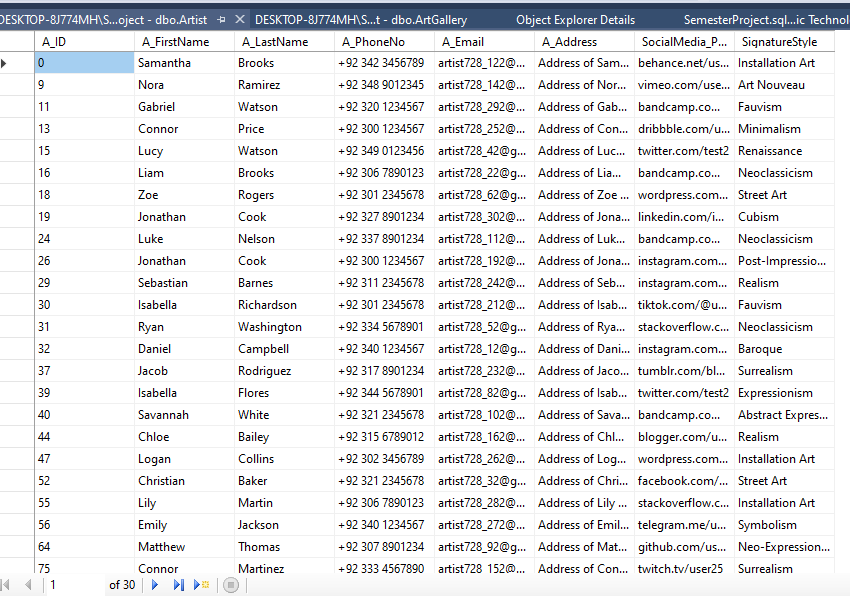
**Client Table:**

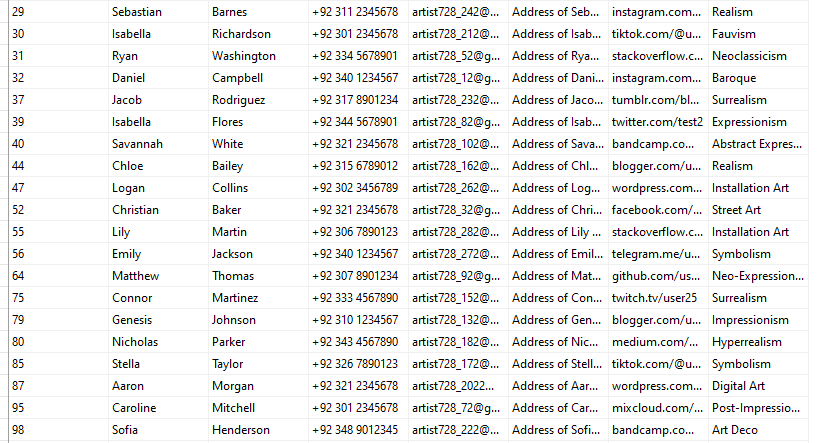
A screenshot of a computer

Description automatically generated

import random  
import pyodbc as odbc  
#List of first\_name  
C\_firstname = ["Aadam", "Aalia", "Aamir", "Aariz", "Aasia", "Abdul", "Abeeha", "Adeel", "Adil", "Afreen",  
 "Afzal", "Ahmed", "Aisha", "Akbar", "Aleeza", "Ali", "Alina", "Alisha", "Aman", "Amina",  
 "Aminah", "Amir", "Ammar", "Amna", "Anaya", "Ansa", "Anwar", "Aqsa", "Arham", "Arif",  
 "Arisha", "Arslan", "Asad", "Asfia", "Ashar", "Ashraf", "Asim", "Asma", "Asrar", "Atif",  
 "Ayesha", "Ayman", "Aysha", "Azam", "Azhar", "Aziz", "Baqir", "Bashir", "Bilal", "Danish",  
 "Daud", "Ehsan", "Faheem", "Fahad", "Faiza", "Faisal", "Faizan", "Farah", "Farhan", "Faris",  
 "Fatima", "Fawad", "Fiza", "Ghazala", "Gulzar", "Habib", "Hadi", "Hafsa", "Hammad", "Hamza",  
 "Haniya", "Haris", "Hassan", "Hiba", "Hira", "Humaira", "Hussain", "Huzefa", "Ibad", "Ibrahim",  
 "Iftikhar", "Ihsan", "Imran", "Inam", "Inayat", "Iqbal", "Irfan", "Ismail", "Jahanzeb",  
 "Jahangir", "Jamal", "Jamil", "Jawad", "Junaid", "Kamran", "Kashif", "Kausar", "Khalid",  
 "Khawar", "Khurram", "Kiran", "Komal", "Lubna", "Mahmood", "Mahnoor", "Majid", "Makbool",  
 "Malik", "Manzar", "Marium", "Mateen", "Mehboob", "Mehreen", "Mian", "Minhaj", "Moin", "Moiz",  
 "Munawar", "Musa", "Mushtaq", "Mustafa", "Nadia", "Naeem", "Nagina", "Naima", "Najam", "Naseer",  
 "Nasir", "Nawab", "Nawaz", "Nazia", "Nazir", "Nida", "Nisar", "Noman", "Noreen", "Noushad",  
 "Numan", "Nusrat", "Obaid", "Omer", "Owais", "Parveen", "Qaisar", "Qamar", "Qasim", "Raheela",  
 "Raheem", "Rahim", "Rais"]  
  
#List of lastnames  
C\_lastname = ["Abbasi", "Afzal", "Ahmad", "Ali", "Amjad", "Anwar", "Asghar", "Aslam", "Awan", "Aziz",  
 "Baig", "Bajwa", "Bhatti", "Butt", "Chaudhry", "Chishti", "Farooqi", "Ghauri", "Gill", "Haider",  
 "Hussain", "Iqbal", "Javed", "Khan", "Khawaja", "Malik", "Mirza", "Nadeem", "Nawaz", "Qureshi",  
 "Rana", "Rao", "Raza", "Rehman", "Saeed", "Saleem", "Siddiqui", "Tariq", "Usmani", "Warsi",  
 "Yousaf", "Zafar", "Zaidi", "Zaman", "Abbasi", "Afzal", "Ahmad", "Ali", "Amjad", "Anwar",  
 "Asghar", "Aslam", "Awan", "Aziz", "Baig", "Bajwa", "Bhatti", "Butt", "Chaudhry", "Chishti",  
 "Farooqi", "Ghauri", "Gill", "Haider", "Hussain", "Iqbal", "Javed", "Khan", "Khawaja", "Malik",  
 "Mirza", "Nadeem", "Nawaz", "Qureshi", "Rana", "Rao", "Raza", "Rehman", "Saeed", "Saleem",  
 "Siddiqui", "Tariq", "Usmani", "Warsi", "Yousaf", "Zafar", "Zaidi", "Zaman"]  
  
#List of Client Address  
C\_address = ["House No. 123, Block A, Model Town",  
 "Flat No. 56, Gulberg III",  
 "Street No. 7, Garden Town",  
 "Plot No. 89, Johar Town",  
 "Apartment No. 34, DHA Phase 6",  
 "House No. 12-B, Gulshan-e-Ravi",  
 "Plot No. 45, Township",  
 "House No. 78, Wapda Town",  
 "Street No. 23, Allama Iqbal Town",  
 "Flat No. 3, Shadman Colony",  
 "House No. 67, Sabzazar Scheme",  
 "Apartment No. 21, Gulshan-e-Lahore",  
 "House No. 9-C, Green Town",  
 "Plot No. 112, Faisal Town",  
 "Street No. 5, Samanabad",  
 "House No. 33, Garden Town",  
 "Flat No. 8, Gulberg II",  
 "Street No. 12, Iqbal Town",  
 "House No. 56, Model Town Extension",  
 "Apartment No. 7, DHA Phase 5",  
 "Plot No. 23, Johar Town Extension",  
 "Street No. 14, Wapda Town",  
 "House No. 90, Allama Iqbal Town",  
 "Flat No. 11, Shadman Colony",  
 "Plot No. 34, Township Extension",  
 "House No. 45-B, Nishtar Colony",  
 "Apartment No. 9, Gulshan-e-Ravi",  
 "House No. 21, Shahdara Town",  
 "Plot No. 7, Model Town",  
 "Street No. 8, Gulberg IV",  
 "Flat No. 4, Johar Town Phase 2",  
 "House No. 78, Defence View Housing Society",  
 "Apartment No. 32, Valencia Housing Society",  
 "House No. 56, Iqbal Avenue",  
 "Plot No. 12, Bahria Town",  
 "Street No. 21, Garden Town",  
 "Flat No. 67, Gulshan-e-Iqbal",  
 "House No. 11-A, Wapda Town Extension",  
 "Plot No. 33, Green Town",  
 "House No. 45, Samanabad Extension",  
 "Apartment No. 8, DHA Phase 1",  
 "House No. 22, Johar Town",  
 "Plot No. 78, Gulberg I",  
 "Street No. 6, Township",  
 "Flat No. 3, Nishtar Colony",  
 "House No. 89, Iqbal Town Extension",  
 "Apartment No. 11, Shadman Colony",  
 "House No. 34, Allama Iqbal Town",  
 "Plot No. 21, Model Town Extension",  
 "Street No. 9, Gulberg III",  
 "Flat No. 22, Garden Town",  
 "House No. 67, Johar Town Phase 1",  
 "Plot No. 45, Wapda Town Extension",  
 "Apartment No. 12, Defence View Housing Society",  
 "House No. 56, Valencia Housing Society",  
 "Plot No. 34, Iqbal Avenue",  
 "Street No. 12, Bahria Town",  
 "Flat No. 21, Garden Town",  
 "House No. 78, Gulshan-e-Iqbal",  
 "Apartment No. 56, Wapda Town Extension",  
 "House No. 11, Model Town",  
 "Plot No. 33, Gulberg IV",  
 "Street No. 8, Johar Town Phase 2",  
 "Flat No. 4, Defence View Housing Society",  
 "House No. 22, Valencia Housing Society",  
 "Plot No. 78, Iqbal Avenue",  
 "Street No. 6, Bahria Town",  
 "Apartment No. 3, Garden Town",  
 "House No. 89, Gulberg I",  
 "Plot No. 21, Township",  
 "Street No. 9, Nishtar Colony",  
 "Flat No. 22, Iqbal Town Extension",  
 "House No. 67, Shadman Colony",  
 "Plot No. 45, Allama Iqbal Town",  
 "Apartment No. 12, Model Town Extension",  
 "House No. 56, Gulberg III",  
 "Plot No. 34, Johar Town",  
 "Street No. 12, Wapda Town Extension",  
 "Flat No. 21, Defence View Housing Society",  
 "House No. 78, Valencia Housing Society",  
 "Apartment No. 56, Iqbal Avenue",  
 "House No. 11, Bahria Town",  
 "Plot No. 33, Garden Town",  
 "Street No. 8, Gulberg IV",  
 "Flat No. 4, Johar Town Phase 2"]  
  
#List of Phone Numbers  
C\_phoneno = ["+92 300 1234567",  
 "+92 301 2345678",  
 "+92 302 3456789",  
 "+92 303 4567890",  
 "+92 304 5678901",  
 "+92 305 6789012",  
 "+92 306 7890123",  
 "+92 307 8901234",  
 "+92 308 9012345",  
 "+92 309 0123456",  
 "+92 310 1234567",  
 "+92 311 2345678",  
 "+92 312 3456789",  
 "+92 313 4567890",  
 "+92 314 5678901",  
 "+92 315 6789012",  
 "+92 316 7890123",  
 "+92 317 8901234",  
 "+92 318 9012345",  
 "+92 319 0123456",  
 "+92 320 1234567",  
 "+92 321 2345678",  
 "+92 322 3456789",  
 "+92 323 4567890",  
 "+92 324 5678901",  
 "+92 325 6789012",  
 "+92 326 7890123",  
 "+92 327 8901234",  
 "+92 328 9012345",  
 "+92 329 0123456",  
 "+92 330 1234567",  
 "+92 331 2345678",  
 "+92 332 3456789",  
 "+92 333 4567890",  
 "+92 334 5678901",  
 "+92 335 6789012",  
 "+92 336 7890123",  
 "+92 337 8901234",  
 "+92 338 9012345",  
 "+92 339 0123456",  
 "+92 340 1234567",  
 "+92 341 2345678",  
 "+92 342 3456789",  
 "+92 343 4567890",  
 "+92 344 5678901",  
 "+92 345 6789012",  
 "+92 346 7890123",  
 "+92 347 8901234",  
 "+92 348 9012345",  
 "+92 349 0123456",  
 "+92 350 1234567",  
 "+92 351 2345678",  
 "+92 352 3456789",  
 "+92 353 4567890",  
 "+92 354 5678901",  
 "+92 355 6789012",  
 "+92 356 7890123",  
 "+92 357 8901234",  
 "+92 358 9012345",  
 "+92 359 0123456",  
 "+92 360 1234567",  
 "+92 361 2345678",  
 "+92 362 3456789",  
 "+92 363 4567890",  
 "+92 364 5678901",  
 "+92 365 6789012",  
 "+92 366 7890123",  
 "+92 367 8901234",  
 "+92 368 9012345",  
 "+92 369 0123456",  
 "+92 370 1234567",  
 "+92 371 2345678",  
 "+92 372 3456789",  
 "+92 373 4567890",  
 "+92 374 5678901",  
 "+92 375 6789012",  
 "+92 376 7890123",  
 "+92 377 8901234",  
 "+92 378 9012345",  
 "+92 379 0123456",  
 "+92 380 1234567",  
 "+92 381 2345678",  
 "+92 382 3456789",  
 "+92 383 4567890",  
 "+92 384 5678901",  
 "+92 385 6789012",  
 "+92 386 7890123",  
 "+92 387 8901234",  
 "+92 388 9012345",  
 "+92 389 0123456"  
 ]  
  
#List of EmailID  
C\_email = ["user1@example.com",  
 "test2@example.com",  
 "mail3@example.com",  
 "hello4@example.com",  
 "world5@example.com",  
 "user6@example.com",  
 "test7@example.com",  
 "mail8@example.com",  
 "hello9@example.com",  
 "world10@example.com",  
 "user11@example.com",  
 "test12@example.com",  
 "mail13@example.com",  
 "hello14@example.com",  
 "world15@example.com",  
 "user16@example.com",  
 "test17@example.com",  
 "mail18@example.com",  
 "hello19@example.com",  
 "world20@example.com",  
 "user21@example.com",  
 "test22@example.com",  
 "mail23@example.com",  
 "hello24@example.com",  
 "world25@example.com",  
 "user26@example.com",  
 "test27@example.com",  
 "mail28@example.com",  
 "hello29@example.com",  
 "world30@example.com",  
 "user31@example.com",  
 "test32@example.com",  
 "mail33@example.com",  
 "hello34@example.com",  
 "world35@example.com",  
 "user36@example.com",  
 "test37@example.com",  
 "mail38@example.com",  
 "hello39@example.com",  
 "world40@example.com",  
 "user41@example.com",  
 "test42@example.com",  
 "mail43@example.com",  
 "hello44@example.com",  
 "world45@example.com",  
 "user46@example.com",  
 "test47@example.com",  
 "mail48@example.com",  
 "hello49@example.com",  
 "world50@example.com",  
 "user51@example.com",  
 "test52@example.com",  
 "mail53@example.com",  
 "hello54@example.com",  
 "world55@example.com",  
 "user56@example.com",  
 "test57@example.com",  
 "mail58@example.com",  
 "hello59@example.com",  
 "world60@example.com",  
 "user61@example.com",  
 "test62@example.com",  
 "mail63@example.com",  
 "hello64@example.com",  
 "world65@example.com",  
 "user66@example.com",  
 "test67@example.com",  
 "mail68@example.com",  
 "hello69@example.com",  
 "world70@example.com",  
 "user71@example.com",  
 "test72@example.com",  
 "mail73@example.com",  
 "hello74@example.com",  
 "world75@example.com",  
 "user76@example.com",  
 "test77@example.com",  
 "mail78@example.com",  
 "hello79@example.com",  
 "world80@example.com",  
 "user81@example.com",  
 "test82@example.com",  
 "mail83@example.com",  
 "hello84@example.com",  
 "world85@example.com",  
 "user86@example.com",  
 "test87@example.com",  
 "mail88@example.com",  
 "hello89@example.com",  
 "world90@example.com",  
 "user91@example.com",  
 "test92@example.com",  
 "mail93@example.com",  
 "hello94@example.com",  
 "world95@example.com",  
 "user96@example.com",  
 "test97@example.com",  
 "mail98@example.com",  
 "hello99@example.com",  
 "world100@example.com"  
 ]  
  
  
def generate\_record(used\_ids):  
 while True:  
 random\_clientid = random.randint(0, 100)  
 if random\_clientid not in used\_ids:  
 break  
 used\_ids.add(random\_clientid)  
 random\_fname = random.choice(C\_firstname)  
 random\_lname = random.choice(C\_lastname)  
 random\_address = random.choice(C\_address)  
 random\_phoneno = random.choice(C\_phoneno)  
 random\_email = random.choice(C\_email)  
 return {"Client ID": random\_clientid, "First Name": random\_fname, "Last Name": random\_lname, "Address": random\_address,  
 "Phone No": random\_phoneno, "Email": random\_email}  
  
# Generate 30 records  
used\_ids = set()  
records = [generate\_record(used\_ids) for \_ in range(30)]  
  
  
# Database connection details  
SERVER\_NAME = r'DESKTOP-8J774MH\SQLEXPRESS'  
DATABASE\_NAME = 'SemesterProject'  
  
# Connection string  
conn\_str = (  
 f'DRIVER={{ODBC Driver 17 for SQL Server}};'  
 f'SERVER={SERVER\_NAME};'  
 f'DATABASE={DATABASE\_NAME};'  
 r'Trusted\_Connection=yes;'  
)  
  
# Establish connection  
conn = odbc.connect(conn\_str)  
cursor = conn.cursor()  
  
cursor.execute('''  
 IF NOT EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Clients')  
 CREATE TABLE Clients  
 (ClientID INT PRIMARY KEY,  
 FirstName NVARCHAR(50),  
 LastName NVARCHAR(50),  
 Address NVARCHAR(255),  
 PhoneNumber NVARCHAR(20),  
 Email NVARCHAR(255))  
 ''')  
  
# Insert records into the table  
for record in records:  
 cursor.execute('''  
 INSERT INTO Clients (ClientID, FirstName, LastName, Address, PhoneNumber, Email)  
 VALUES (?, ?, ?, ?, ?, ?)  
 ''',  
 record["Client ID"], record["First Name"], record["Last Name"],  
 record["Address"], record["Phone No"], record["Email"])  
 print("Inserted Successfully")  
  
# Commit the transaction  
conn.commit()  
  
# Close the cursor and connection  
cursor.close()  
conn.close()

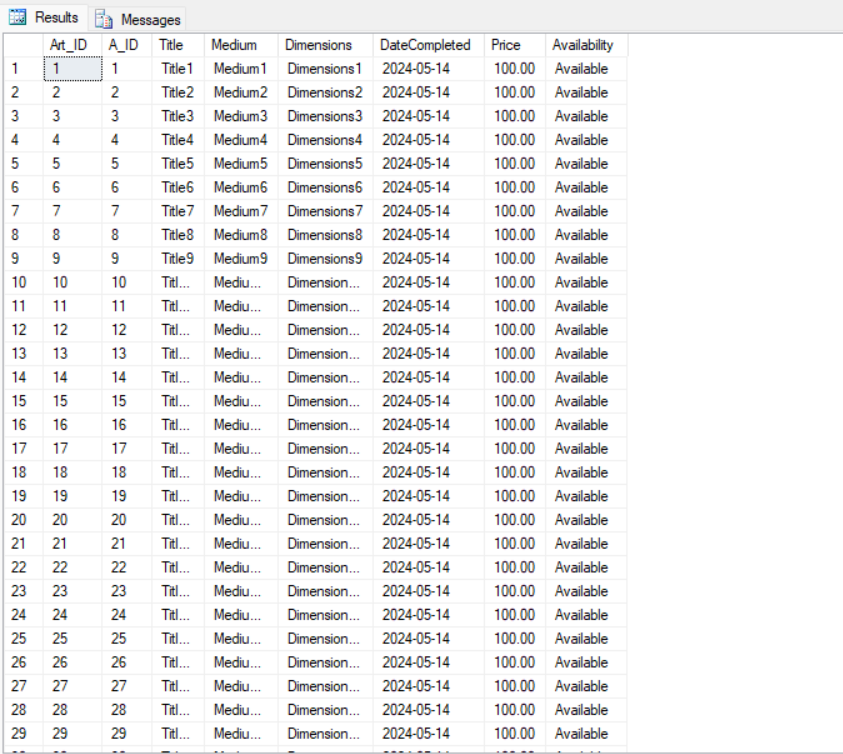
**Artist Table:**





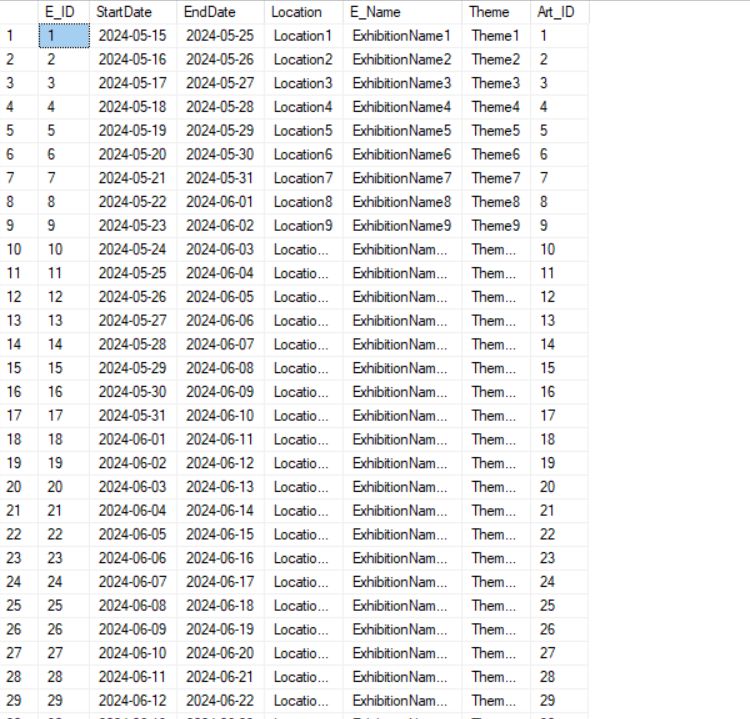
from faker import Faker  
import random  
import pyodbc  
  
# Connect to the SQL Server database  
conn = pyodbc.connect('DRIVER={SQL Server};SERVER=(local);DATABASE=ArtGallery;Trusted\_Connection=yes;')  
cursor = conn.cursor()  
  
# Instantiate Faker  
fake = Faker()  
  
# Generate data for Login table  
for i in range(10):  
 email = fake.email()  
 username = fake.user\_name()  
 cursor.execute("INSERT INTO Login (Login\_ID, Email, Username) VALUES (?, ?, ?)",  
 (i+1, email, username))  
  
# Generate data for Staff table  
for i in range(10):  
 s\_address = fake.address()  
 s\_firstname = fake.first\_name()  
 s\_lastname = fake.last\_name()  
 s\_email = fake.email()  
 s\_phone = fake.phone\_number()  
 cursor.execute("INSERT INTO Staff (S\_ID, S\_Address, S\_FirstName, S\_LastName, S\_Email, S\_PhoneNo) VALUES (?, ?, ?, ?, ?, ?)",  
 (i+1, s\_address, s\_firstname, s\_lastname, s\_email, s\_phone))  
  
# Generate data for Client table  
for i in range(10):  
 c\_email = fake.email()  
 c\_firstname = fake.first\_name()  
 c\_lastname = fake.last\_name()  
 c\_phone = fake.phone\_number()  
 c\_address = fake.address()  
 cursor.execute("INSERT INTO Client (C\_ID, C\_Email, C\_FirstName, C\_LastName, C\_PhoneNo, C\_Address) VALUES (?, ?, ?, ?, ?, ?)",  
 (i+1, c\_email, c\_firstname, c\_lastname, c\_phone, c\_address))  
  
# Generate data for Artist table  
for i in range(10):  
 a\_firstname = fake.first\_name()  
 a\_lastname = fake.last\_name()  
 a\_phone = fake.phone\_number()  
 a\_email = fake.email()  
 a\_address = fake.address()  
 social\_media = fake.url()  
 signature\_style = fake.text()  
 cursor.execute("INSERT INTO Artist (A\_ID, A\_FirstName, A\_LastName, A\_PhoneNo, A\_Email, A\_Address, SocialMedia\_Profile, SignatureStyle) VALUES (?, ?, ?, ?, ?, ?, ?, ?)",  
 (i+1, a\_firstname, a\_lastname, a\_phone, a\_email, a\_address, social\_media, signature\_style))  
  
# Generate data for Artwork table  
for i in range(10):  
 title = fake.sentence(nb\_words=3)  
 medium = fake.word()  
 dimensions = fake.random\_int(min=1, max=1000)  
 date\_completed = fake.date\_this\_century()  
 price = fake.random\_number(digits=5)  
 availability = random.choice(["Available", "Sold", "On Loan"])  
 a\_id = random.randint(1, 10) # Assuming there are 10 artists  
 cursor.execute("INSERT INTO Artwork (Art\_ID, A\_ID, Title, Medium, Dimensions, DateCompleted, Price, Availability) VALUES (?, ?, ?, ?, ?, ?, ?, ?)",  
 (i+1, a\_id, title, medium, dimensions, date\_completed, price, availability))  
  
# Generate data for Exhibition table  
for i in range(10):  
 start\_date = fake.date\_this\_year()  
 end\_date = fake.date\_between\_dates(date\_start=start\_date)  
 location = fake.address()  
 e\_name = fake.word()  
 theme = fake.sentence(nb\_words=3)  
 art\_id = random.randint(1, 10) # Assuming there are 10 artworks  
 cursor.execute("INSERT INTO Exhibition (E\_ID, StartDate, EndDate, Location, E\_Name, Theme, Art\_ID) VALUES (?, ?, ?, ?, ?, ?, ?)",  
 (i+1, start\_date, end\_date, location, e\_name, theme, art\_id))  
  
# Generate data for Gallery table  
for i in range(10):  
 gallery\_name = fake.company()  
 status = random.choice(["Open", "Closed"])  
 location = fake.address()  
 cursor.execute("INSERT INTO Gallery (G\_ID, GalleryName, Status, Location) VALUES (?, ?, ?, ?)",  
 (i+1, gallery\_name, status, location))  
  
# Generate data for Sales table  
for i in range(10):  
 art\_id = random.randint(1, 10) # Assuming there are 10 artworks  
 c\_id = random.randint(1, 10) # Assuming there are 10 clients  
 sale\_price = fake.random\_number(digits=5)  
 sale\_date = fake.date\_this\_year()  
 artist\_commission = sale\_price \* 0.2 # Assuming 20% commission for the artist  
 cursor.execute("INSERT INTO Sales (Sale\_ID, Art\_ID, C\_ID, SalePrice, SaleDate, ArtistCommision) VALUES (?, ?, ?, ?, ?, ?)",  
 (i+1, art\_id, c\_id, sale\_price, sale\_date, artist\_commission))  
  
# Generate data for Loan table  
for i in range(10):  
 art\_id = random.randint(1, 10) # Assuming there are 10 artworks  
 loan\_agreement = fake.sentence(nb\_words=5)  
 duration = random.randint(1, 12)  
 expiry\_date = fake.date\_between(start\_date='today', end\_date='+1y')  
 transport\_tracking = fake.random\_int(min=1000, max=9999)  
 monitoring\_status = random.choice(["Active", "Inactive"])  
 cursor.execute("INSERT INTO Loan (Loan\_ID, Art\_ID, LoanAgreement, Duration, ExpiryDate, TransportTracking, MonitoringStatus) VALUES (?, ?, ?, ?, ?, ?, ?)",  
 (i+1, art\_id, loan\_agreement, duration, expiry\_date, transport\_tracking, monitoring\_status))  
conn.commit()  
  
# Close the cursor and connection  
cursor.close()  
conn.close()

**Artwork Table:**

****

import random  
import pyodbc  
from datetime import datetime, timedelta  
  
# Database connection details  
SERVER\_NAME = r'DESKTOP-L5NG8PP\SQLEXPRESS'  
DATABASE\_NAME = 'ArtGallery'  
  
# Define connection string  
conn\_str = (  
 f'DRIVER={{ODBC Driver 17 for SQL Server}};'  
 f'SERVER={SERVER\_NAME};'  
 f'DATABASE={DATABASE\_NAME};'  
 r'Trusted\_Connection=yes;' # For Windows Authentication  
)  
  
try:  
 # Connect to the SQL Server  
 conn = pyodbc.connect(conn\_str)  
 cursor = conn.cursor()  
  
 # Create the Sales table if it doesn't exist  
 cursor.execute('''  
 IF NOT EXISTS (SELECT \* FROM sys.tables WHERE name = 'Sales')  
 CREATE TABLE Sales (  
 Sale\_ID INT PRIMARY KEY,  
 Art\_ID INT,  
 C\_ID INT,  
 SalePrice DECIMAL(10, 2),  
 SaleDate DATE,  
 ArtistCommision DECIMAL(10, 2),  
 FOREIGN KEY (Art\_ID) REFERENCES Artwork(Art\_ID),  
 FOREIGN KEY (C\_ID) REFERENCES Client(C\_ID)  
 )  
 ''')  
 conn.commit()  
  
 # Function to generate random sale date  
 def generate\_sale\_date():  
 start\_date = datetime(2010, 1, 1)  
 end\_date = datetime.now()  
 duration = (end\_date - start\_date).days  
 random\_days = random.randint(0, duration)  
 sale\_date = start\_date + timedelta(days=random\_days)  
 return sale\_date  
  
 # Generate and insert 30 random records  
 for \_ in range(30):  
 sale\_id = random.randint(1000, 9999) # Assuming sale\_id is a 4-digit number  
 art\_id = random.randint(0, 100) # Assuming art\_id ranges from 0 to 100  
 c\_id = random.randint(1, 100) # Assuming c\_id ranges from 1 to 100  
 sale\_price = round(random.uniform(100, 10000), 2)  
 sale\_date = generate\_sale\_date().strftime('%Y-%m-%d') # Format sale date as YYYY-MM-DD  
 artist\_commission = round(sale\_price \* 0.1, 2) # Assuming artist commission is 10% of sale price  
  
 # Check if the art\_id exists in the Artwork table  
 cursor.execute("SELECT COUNT(\*) FROM Artwork WHERE Art\_ID = ?", art\_id)  
 if cursor.fetchone()[0] > 0:  
 # Insert data into SQL table  
 cursor.execute("INSERT INTO Sales (Sale\_ID, Art\_ID, C\_ID, SalePrice, SaleDate, ArtistCommision) VALUES (?, ?, ?, ?, ?, ?)",  
 (sale\_id, art\_id, c\_id, sale\_price, sale\_date, artist\_commission))  
 conn.commit()  
 else:  
 print(f"Artwork with ID {art\_id} does not exist.")  
  
 print("Data inserted successfully.")  
  
except pyodbc.Error as e:  
 print("An error occurred:", e)  
  
finally:  
 # Close connection  
 conn.close()

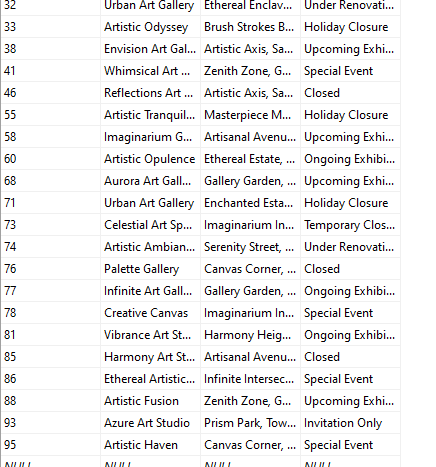
**Exhibition Table:**

****

import random  
import pyodbc as odbc  
from datetime import datetime, timedelta  
  
# Assuming you have defined E\_name, E\_location, and artwork\_themes somewhere  
  
def generate\_random\_dates(start\_year, end\_year):  
 # Generate a random year between start\_year and end\_year  
 year = random.randint(start\_year, end\_year)  
 month = random.randint(1, 12)  
 day = random.randint(1, 28)  
  
 # Create start date  
 start\_date = datetime(year, month, day)  
  
 # Generate a random duration between 1 and 365 days  
 duration = random.randint(1, 365)  
  
 # Calculate end date by adding duration to start date  
 end\_date = start\_date + timedelta(days=duration)  
  
 return start\_date, end\_date  
  
def generate\_record(artwork\_ids):  
 random\_exhibitionid = random.randint(0, 100)  
 random\_ename = random.choice(E\_name)  
 random\_elocation = random.choice(E\_location)  
 random\_themes = random.choice(artwork\_themes)  
 random\_artworkid = random.choice(artwork\_ids)  
  
 # Call the generate\_random\_dates function to get start and end dates  
 start\_date, end\_date = generate\_random\_dates(2022, 2023)  
  
 return (random\_exhibitionid, random\_ename, random\_elocation, random\_themes,  
 start\_date.strftime("%Y-%m-%d"), end\_date.strftime("%Y-%m-%d"), random\_artworkid)  
  
# Generate artwork IDs  
artwork\_ids = [i for i in range(1, 101)] # Assuming 100 artworks  
  
# Generate 30 records  
records = [generate\_record(artwork\_ids) for \_ in range(30)]  
  
# Connect to SQL Server  
SERVER\_NAME = r'DESKTOP-L5NG8PP\SQLEXPRESS'  
DATABASE\_NAME = 'ArtGallery'  
  
# Define connection string  
conn\_str = (  
 f'DRIVER={{ODBC Driver 17 for SQL Server}};'  
 f'SERVER={SERVER\_NAME};'  
 f'DATABASE={DATABASE\_NAME};'  
 r'Trusted\_Connection=yes;' # For Windows Authentication  
)  
conn = odbc.connect(conn\_str)  
cursor = conn.cursor()  
  
# Insert records into SQL tables  
for record in records:  
 sql\_query = '''  
 INSERT INTO Exhibitions (ExhibitionID, ExhibitionName, ExhibitionLocation, Themes, StartDate, EndDate)  
 VALUES (?, ?, ?, ?, ?, ?)  
 '''  
  
print("Inserted Successfully")  
# Commit the transaction  
conn.commit()  
  
# Close the connection  
conn.close()

**Gallery Table:**





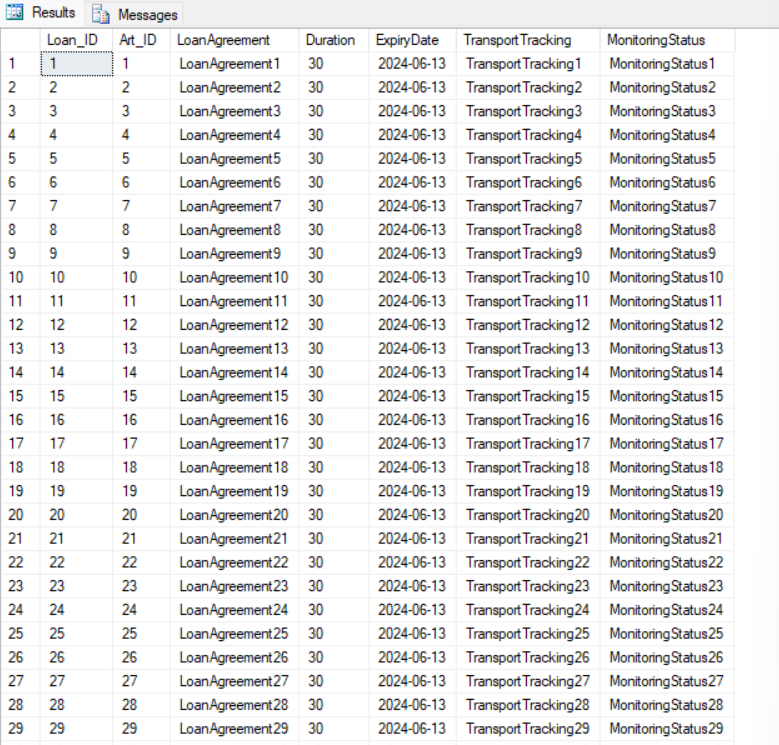
import pyodbc as odbc  
import random  
  
# List of Art Gallery Names  
art\_gallery = ["Gallery Glimpse",  
 "Art Haven",  
 "Visionary Art Gallery",  
 "Creative Canvas",  
 "Expressive Art Space",  
 "Artistic Sanctuary",  
 "Spectrum Gallery",  
 "Brush Strokes Gallery",  
 "Masterpiece Gallery",  
 "Artistic Impressions",  
 "Muse Art Gallery",  
 "Canvas Creations",  
 "Serenity Art Space",  
 "Palette Gallery",  
 "Artful Endeavors",  
 "Dreamy Art Gallery",  
 "Harmony Art Studio",  
 "Artistic Fusion",  
 "Urban Art Gallery",  
 "Ethereal Art Space",  
 "Colorful Creations Gallery",  
 "Tranquil Art Haven",  
 "Inspire Art Gallery",  
 "Renaissance Art Studio",  
 "Creative Edge Gallery",  
 "Infinite Art Gallery",  
 "Artistic Evolution",  
 "Gallery Elegance",  
 "Vibrant Art Space",  
 "Envision Art Gallery",  
 "Artisan Showcase",  
 "Prism Gallery",  
 "Artistic Journey Gallery",  
 "Tranquility Art Studio",  
 "Artistic Visions",  
 "Whimsical Art Gallery",  
 "Unity Art Space",  
 "Cosmic Art Gallery",  
 "Wonderland Art Studio",  
 "Radiant Art Gallery",  
 "Reflections Art Space",  
 "Phoenix Art Gallery",  
 "Elemental Art Studio",  
 "Aurora Art Gallery",  
 "Artistic Expressions",  
 "Luminous Art Space",  
 "Essence Art Gallery",  
 "Gallery Odyssey",  
 "Azure Art Studio",  
 "Harmony Art Gallery",  
 "Artistic Oasis",  
 "Dreamland Art Space",  
 "Fusion Art Gallery",  
 "Celestial Art Studio",  
 "Artistic Euphoria",  
 "Imaginarium Gallery",  
 "Serendipity Art Space",  
 "Essence Art Studio",  
 "Artistic Horizons",  
 "Gallery Genesis",  
 "Zenith Art Gallery",  
 "Artistic Muse",  
 "Ethereal Art Studio",  
 "Sanctuary Art Space",  
 "Infinite Artistry Gallery",  
 "Artistic Escape",  
 "Enchanted Art Gallery",  
 "Gallery Elysium",  
 "Artistic Odyssey",  
 "Celestial Art Space",  
 "Artisanal Art Studio",  
 "Reflections Art Gallery",  
 "Serenity Artistic Space",  
 "Artistic Transcendence",  
 "Gallery Utopia",  
 "Canvas Dreams Art Studio",  
 "Visionary Artistic Space",  
 "Artistic Paradise Gallery",  
 "Artistic Reverie",  
 "Zen Art Space",  
 "Whimsy Art Gallery",  
 "Artistic Elegance",  
 "Gallery of Dreams",  
 "Envision Artistic Space",  
 "Kaleidoscope Art Studio",  
 "Artistic Radiance",  
 "Artistic Reverence Gallery",  
 "Ethereal Artistic Space",  
 "Gallery of Wonders",  
 "Artistic Haven",  
 "Vibrance Art Studio",  
 "Artistic Ambiance",  
 "Gallery Nirvana",  
 "Artistic Opulence",  
 "Whimsical Artistic Space",  
 "Artistic Tranquility",  
 "Gallery Alchemy",  
 "Artistic Nirvana",  
 "Muse Artistic Space",  
 "Artistic Serenity Gallery"  
 ]  
  
# List of ArtGallery Locations  
G\_location = ["123 Art Street, Lahore",  
 "Artistic Avenue, Model Town, Lahore",  
 "Creative Circle, Gulberg, Lahore",  
 "Canvas Corner, Johar Town, Lahore",  
 "Expressive Lane, DHA, Lahore",  
 "Palette Place, Garden Town, Lahore",  
 "Spectrum Square, Lahore Cantt",  
 "Brush Strokes Boulevard, Gulshan-e-Iqbal, Lahore",  
 "Masterpiece Mansion, Allama Iqbal Town, Lahore",  
 "Inspire Plaza, Faisal Town, Lahore",  
 "Muse Market, Bahria Town, Lahore",  
 "Harmony Heights, Ichra, Lahore",  
 "Vibrant Venue, Defence Road, Lahore",  
 "Tranquil Terrace, Wapda Town, Lahore",  
 "Artistic Axis, Samanabad, Lahore",  
 "Dreamy Drive, Shahdara, Lahore",  
 "Artisan Avenue, Model Town Link Road, Lahore",  
 "Prism Park, Township, Lahore",  
 "Whimsical Walk, Raiwind Road, Lahore",  
 "Cosmic Corner, Thokar Niaz Baig, Lahore",  
 "Creative Crescent, Bedian Road, Lahore",  
 "Artful Alley, Gulshan Ravi, Lahore",  
 "Inspire Street, Sabzazar, Lahore",  
 "Renaissance Road, Garden Town, Lahore",  
 "Gallery Garden, Johar Town, Lahore",  
 "Tranquility Tower, Lahore Cantt",  
 "Serenity Street, Cavalry Ground, Lahore",  
 "Artisanal Avenue, Gulberg III, Lahore",  
 "Ethereal Estate, Model Town, Lahore",  
 "Imaginarium Intersection, DHA Phase 5, Lahore",  
 "Zenith Zone, Gulshan-e-Ravi, Lahore",  
 "Sanctuary Street, Defence Housing Society, Lahore",  
 "Infinite Intersection, Township, Lahore",  
 "Artistic Axis, Samanabad, Lahore",  
 "Enchanted Estate, Wapda Town, Lahore",  
 "Gallery Garden, Johar Town, Lahore",  
 "Muse Mansion, Gulshan-e-Iqbal, Lahore",  
 "Ethereal Enclave, Model Town Link Road, Lahore",  
 "Tranquility Tower, Lahore Cantt",  
 "Serenity Street, Cavalry Ground, Lahore",  
 "Artisanal Avenue, Gulberg III, Lahore",  
 "Ethereal Estate, Model Town, Lahore",  
 "Imaginarium Intersection, DHA Phase 5, Lahore",  
 "Zenith Zone, Gulshan-e-Ravi, Lahore",  
 "Sanctuary Street, Defence Housing Society, Lahore",  
 "Infinite Intersection, Township, Lahore",  
 "Artistic Axis, Samanabad, Lahore",  
 "Enchanted Estate, Wapda Town, Lahore",  
 "Gallery Garden, Johar Town, Lahore",  
 "Muse Mansion, Gulshan-e-Iqbal, Lahore",  
 "Ethereal Enclave, Model Town Link Road, Lahore",  
 "Tranquility Tower, Lahore Cantt",  
 "Serenity Street, Cavalry Ground, Lahore",  
 "Artisanal Avenue, Gulberg III, Lahore",  
 "Ethereal Estate, Model Town, Lahore",  
 "Imaginarium Intersection, DHA Phase 5, Lahore",  
 "Zenith Zone, Gulshan-e-Ravi, Lahore",  
 "Sanctuary Street, Defence Housing Society, Lahore",  
 "Infinite Intersection, Township, Lahore",  
 "Artistic Axis, Samanabad, Lahore",  
 "Enchanted Estate, Wapda Town, Lahore",  
 "Gallery Garden, Johar Town, Lahore",  
 "Muse Mansion, Gulshan-e-Iqbal, Lahore",  
 "Ethereal Enclave, Model Town Link Road, Lahore",  
 "Tranquility Tower, Lahore Cantt",  
 "Serenity Street, Cavalry Ground, Lahore",  
 "Artisanal Avenue, Gulberg III, Lahore",  
 "Ethereal Estate, Model"  
 ]  
  
# List of Status of Art Gallery  
G\_status = ["Open",  
 "Closed",  
 "Under Renovation",  
 "Upcoming Exhibition",  
 "Ongoing Exhibition",  
 "Sold Out",  
 "Invitation Only",  
 "Special Event",  
 "Holiday Closure",  
 "Temporary Closure"  
 ]  
  
  
def generate\_record(used\_ids):  
 while True:  
 random\_galleryid = random.randint(0, 100)  
 if random\_galleryid not in used\_ids:  
 break  
 used\_ids.add(random\_galleryid)  
 random\_galleryname = random.choice(art\_gallery)  
 random\_gallerylocation = random.choice(G\_location)  
 random\_status = random.choice(G\_status)  
 return {"Gallery ID": random\_galleryid, "Gallery Name": random\_galleryname,  
 "Gallery Location": random\_gallerylocation, "Gallery Status": random\_status}  
  
# Generate 30 records  
used\_ids = set()  
records = [generate\_record(used\_ids) for \_ in range(30)]  
  
  
SERVER\_NAME = r'DESKTOP-8J774MH\SQLEXPRESS'  
DATABASE\_NAME = 'SemesterProject'  
  
# Define connection string  
conn\_str = (  
 f'DRIVER={{ODBC Driver 17 for SQL Server}};'  
 f'SERVER={SERVER\_NAME};'  
 f'DATABASE={DATABASE\_NAME};'  
 r'Trusted\_Connection=yes;' # For Windows Authentication  
)  
conn = odbc.connect(conn\_str)  
# Create a cursor  
cursor = conn.cursor()  
  
# Create a table  
cursor.execute('''  
 CREATE TABLE ArtGallery  
 (GalleryID INT PRIMARY KEY,  
 GalleryName NVARCHAR(255),  
 GalleryLocation NVARCHAR(255),  
 GalleryStatus NVARCHAR(50))  
 ''')  
  
# Insert data into the table  
for record in records:  
 cursor.execute('''  
 INSERT INTO ArtGallery (GalleryID, GalleryName, GalleryLocation, GalleryStatus)  
 VALUES (?, ?, ?, ?)  
 ''',  
 record["Gallery ID"], record["Gallery Name"], record["Gallery Location"], record["Gallery Status"])  
 print("Inserted Successfully")  
# Commit the transaction  
conn.commit()  
  
# Close the cursor and connection  
cursor.close()  
conn.close()

**Sales Table:**

****

import random  
import pyodbc as odbc  
from faker import Faker  
  
fake = Faker()  
  
def generate\_staff\_record(used\_ids):  
 while True:  
 random\_staff\_id = random.randint(1, 1000)  
 if random\_staff\_id not in used\_ids:  
 break  
 used\_ids.add(random\_staff\_id)  
 random\_address = fake.address()  
 random\_first\_name = fake.first\_name()  
 random\_last\_name = fake.last\_name()  
 random\_email = fake.email()  
 random\_phone\_no = fake.phone\_number() # Generate phone number without extension  
 return {"S\_ID": random\_staff\_id, "S\_Address": random\_address, "S\_FirstName": random\_first\_name, "S\_LastName": random\_last\_name,  
 "S\_Email": random\_email, "S\_PhoneNo": random\_phone\_no}  
  
# Generate 30 staff records  
used\_ids = set()  
staff\_records = [generate\_staff\_record(used\_ids) for \_ in range(30)]  
  
# Database connection details  
SERVER\_NAME = r'DESKTOP-L5NG8PP\SQLEXPRESS'  
DATABASE\_NAME = 'ArtGallery'  
  
# Connection string  
conn\_str = (  
 f'DRIVER={{ODBC Driver 17 for SQL Server}};'  
 f'SERVER={SERVER\_NAME};'  
 f'DATABASE={DATABASE\_NAME};'  
 r'Trusted\_Connection=yes;'  
)  
  
# Establish connection  
conn = odbc.connect(conn\_str)  
cursor = conn.cursor()  
  
# Create Staff table if it doesn't exist  
cursor.execute('''  
 IF NOT EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Staff')  
 CREATE TABLE Staff  
 (S\_ID INT PRIMARY KEY,  
 S\_Address VARCHAR(255),  
 S\_FirstName VARCHAR(100),  
 S\_LastName VARCHAR(100),  
 S\_Email VARCHAR(255) UNIQUE,  
 S\_PhoneNo VARCHAR(20))  
 ''')  
  
# Insert records into the Staff table  
for record in staff\_records:  
 cursor.execute('''  
 INSERT INTO Staff (S\_ID, S\_Address, S\_FirstName, S\_LastName, S\_Email, S\_PhoneNo)  
 VALUES (?, ?, ?, ?, ?, ?)  
 ''',  
 record["S\_ID"], record["S\_Address"], record["S\_FirstName"], record["S\_LastName"],  
 record["S\_Email"], record["S\_PhoneNo"])  
 print("Inserted Successfully")  
  
# Commit the transaction  
conn.commit()  
  
# Close the cursor and connection  
cursor.close()  
conn.close()

**Loan Table:**

****

import random  
import pyodbc  
from datetime import datetime, timedelta  
  
# Database connection details  
SERVER\_NAME = r'DESKTOP-8J774MH\SQLEXPRESS'  
DATABASE\_NAME = 'SemesterProject'  
  
# Define connection string  
conn\_str = (  
 f'DRIVER={{ODBC Driver 17 for SQL Server}};'  
 f'SERVER={SERVER\_NAME};'  
 f'DATABASE={DATABASE\_NAME};'  
 r'Trusted\_Connection=yes;' # For Windows Authentication  
)  
  
try:  
 # Connect to the SQL Server  
 conn = pyodbc.connect(conn\_str)  
 cursor = conn.cursor()  
  
 # Create the LOAN table if it doesn't exist  
 cursor.execute('''  
 IF NOT EXISTS (SELECT \* FROM sys.tables WHERE name = 'LOAN')  
 CREATE TABLE LOAN (  
 loan\_id INT PRIMARY KEY,  
 artwork\_id INT,  
 loan\_agreement\_status VARCHAR(3),  
 transport\_tracking VARCHAR(50),  
 monitoring\_status VARCHAR(8),  
 expiry\_date DATE  
 )  
 ''')  
 conn.commit()  
  
 # Define lists for attributes  
 loan\_agreement = ['yes', 'no']  
 transport\_track = ['Leapords', 'DHL Express', 'FedEX', 'TCS', 'Pakistan Post']  
 monitoring\_status = ['borrowed', 'returned']  
  
 # Function to generate random expiry date  
 def generate\_expiry\_date():  
 start\_date = datetime.now()  
 duration = random.randint(1, 365)  
 expiry\_date = start\_date + timedelta(days=duration)  
 return expiry\_date  
  
 # Generate and insert 30 random records  
 for \_ in range(30):  
 loan\_id = random.randint(1000, 9999) # Assuming loan\_id is a 4-digit number  
 artwork\_id = random.randint(1, 100) # Assuming artwork\_id ranges from 1 to 100  
 loan\_agreement\_status = random.choice(loan\_agreement)  
 transport\_tracking = random.choice(transport\_track)  
 monitoring\_status = random.choice(monitoring\_status)  
 expiry\_date = generate\_expiry\_date().strftime('%Y-%m-%d') # Format expiry date as YYYY-MM-DD  
  
 # Insert data into SQL table  
 cursor.execute("INSERT INTO LOAN (loan\_id, artwork\_id, loan\_agreement\_status, transport\_tracking, monitoring\_status, expiry\_date) VALUES (?, ?, ?, ?, ?, ?)",  
 (loan\_id, artwork\_id, loan\_agreement\_status, transport\_tracking, monitoring\_status, expiry\_date))  
 conn.commit()  
  
 print("Data inserted successfully.")  
  
except pyodbc.Error as e:  
 print("An error occurred:", e)  
  
finally:  
 # Close connection  
 conn.close()