**University of Central Punjab**

**Department of Computer Science**

**Project Report – Semester Spring 2024**

**Project Title: BuildingPro**

**Submitted By:**

* Ayaan Zaman Khattak (0224)
* M Zafar Ul Haq (0193)
* Abdul Moeiz Majid Khan (0225)

**Submitted To:**

* [Maam Saira Shairi]

**Abstract**

BuildingPro is an innovative software solution designed to streamline the management and operations of building projects. This system aims to enhance efficiency, improve communication, and ensure effective project tracking and resource management. By integrating modern technology, BuildingPro provides a comprehensive platform that supports various aspects of construction management, including project planning, scheduling, budgeting, and documentation.

**Introduction**

The construction industry is often plagued by inefficiencies and communication breakdowns, leading to delays and increased costs. BuildingPro addresses these challenges by offering a centralized platform that brings together all stakeholders, from project managers to subcontractors, ensuring seamless collaboration and real-time information sharing.

**Objectives:**

1. Develop a user-friendly interface for project management.
2. Implement robust features for scheduling and resource allocation.
3. Ensure real-time tracking and reporting capabilities.
4. Enhance communication among team members and stakeholders.
5. Integrate financial management tools to monitor budgets and expenses.

**System Design**

**Architecture:** BuildingPro employs a multi-tier architecture comprising a user interface layer, an application logic layer, and a data storage layer. This design ensures scalability, maintainability, and ease of integration with other systems.

**Modules:**

1. **Project Management:** Tools for creating, updating, and tracking project milestones and tasks.
2. **Resource Management:** Allocation and monitoring of resources, including labor, materials, and equipment.
3. **Scheduling:** Gantt charts and calendar views for planning and adjusting project timelines.
4. **Financial Management:** Budget tracking, expense monitoring, and financial reporting.
5. **Communication:** Messaging and notification systems to keep all team members informed.

**Methodology**

The development of BuildingPro followed the Agile methodology, allowing for iterative progress and continuous feedback from stakeholders. This approach ensured that the final product was closely aligned with user needs and industry requirements.

**Development Tools:**

* Programming Languages: JavaScript, Python
* Frameworks: React for front-end development, Django for back-end development
* Database: PostgreSQL
* Project Management: Jira

**Features**

**User Interface:** BuildingPro boasts a clean and intuitive user interface, designed to be accessible to users with varying levels of technical expertise. The dashboard provides an overview of active projects, upcoming deadlines, and key performance indicators.

**Scheduling and Planning:** The system includes advanced scheduling tools that allow project managers to create detailed project plans, assign tasks, and adjust timelines as needed. Gantt charts provide a visual representation of project progress and dependencies.

**Resource Management:** Resource allocation is simplified through a dedicated module that tracks the availability and utilization of materials, labor, and equipment. Alerts are generated when resources are over-allocated or under-utilized.

**Financial Management:** BuildingPro includes comprehensive financial management tools that enable users to create budgets, track expenses, and generate financial reports. This ensures that projects remain within budget and financial health is continuously monitored.

**Communication:** The platform supports robust communication features, including messaging, email notifications, and document sharing. This ensures that all stakeholders are kept informed and can collaborate effectively.

**Implementation**

The implementation phase involved rigorous testing to ensure the system's reliability and performance. User acceptance testing (UAT) was conducted with a select group of industry professionals to gather feedback and make necessary adjustments.

**Deployment:** BuildingPro is deployed on a cloud-based infrastructure, providing scalability and accessibility. Regular updates and maintenance are scheduled to ensure the system remains up-to-date with the latest features and security patches.

**Conclusion**

BuildingPro represents a significant advancement in construction project management, offering a comprehensive and user-friendly platform that addresses the industry's most pressing challenges. The system's robust features and seamless integration capabilities make it an invaluable tool for project managers and stakeholders.

**Future Work:** Future enhancements will focus on incorporating advanced analytics and machine learning to provide predictive insights and further optimize project management processes.

**PROJECT 2 INSERTION AND JOINS**

CREATE DATABASE RealEstateManagementSystem;

USE RealEstateManagementSystem;

CREATE TABLE Clients (

ClientID INT AUTO\_INCREMENT PRIMARY KEY,

FirstName VARCHAR(100),

LastName VARCHAR(100),

Email VARCHAR(255),

Phone VARCHAR(20)

);

CREATE TABLE Landlords (

LandlordID INT AUTO\_INCREMENT PRIMARY KEY,

FirstName VARCHAR(100),

LastName VARCHAR(100),

Email VARCHAR(255),

Phone VARCHAR(20)

);

CREATE TABLE Properties (

PropertyID INT AUTO\_INCREMENT PRIMARY KEY,

Address VARCHAR(255),

City VARCHAR(100),

State VARCHAR(100),

PostalCode VARCHAR(20),

Country VARCHAR(100),

OwnerID INT,

FOREIGN KEY (OwnerID) REFERENCES Landlords(LandlordID)

);

CREATE TABLE Listings (

ListingID INT AUTO\_INCREMENT PRIMARY KEY,

PropertyID INT,

ListingType ENUM('Sale', 'Rent'),

Price DECIMAL(10, 2),

ListingDate DATE,

Status ENUM('Available', 'Sold', 'Rented'),

FOREIGN KEY (PropertyID) REFERENCES Properties(PropertyID)

);

CREATE TABLE Sales (

SaleID INT AUTO\_INCREMENT PRIMARY KEY,

ListingID INT,

ClientID INT,

SaleDate DATE,

SalePrice DECIMAL(10, 2),

FOREIGN KEY (ListingID) REFERENCES Listings(ListingID),

FOREIGN KEY (ClientID) REFERENCES Clients(ClientID)

);

CREATE TABLE Rents (

RentID INT AUTO\_INCREMENT PRIMARY KEY,

ListingID INT,

ClientID INT,

StartDate DATE,

EndDate DATE,

RentAmount DECIMAL(10, 2),

FOREIGN KEY (ListingID) REFERENCES Listings(ListingID),

FOREIGN KEY (ClientID) REFERENCES Clients(ClientID)

);

CREATE TABLE Payments (

PaymentID INT AUTO\_INCREMENT PRIMARY KEY,

ClientID INT,

RentID INT,

Amount DECIMAL(10, 2),

PaymentDate DATE,

PaymentMethod VARCHAR(100),

FOREIGN KEY (ClientID) REFERENCES Clients(ClientID),

FOREIGN KEY (RentID) REFERENCES Rents(RentID)

);

CREATE TABLE MaintenanceRequests (

RequestID INT AUTO\_INCREMENT PRIMARY KEY,

PropertyID INT,

ClientID INT,

Description TEXT,

RequestDate DATE,

Status ENUM('Pending', 'In Progress', 'Completed'),

FOREIGN KEY (PropertyID) REFERENCES Properties(PropertyID),

FOREIGN KEY (ClientID) REFERENCES Clients(ClientID)

);

**INSERTION**

-- Inserting data into Landlords

INSERT INTO Landlords (FirstName, LastName, Email, Phone)

VALUES

('Ahmed', 'Khan', 'ahmed.khan@example.com', '+923001234567'),

('Fatima', 'Malik', 'fatima.malik@example.com', '+923006543210'),

('Sajid', 'Butt', 'sajid.butt@example.com', '+923212345678'),

('Zainab', 'Hussain', 'zainab.hussain@example.com', '+923213456789'),

('Bilal', 'Ahmed', 'bilal.ahmed@example.com', '+923314567890'),

('Imran', 'Sheikh', 'imran.sheikh@example.com', '+923325678901'),

('Maria', 'Akhtar', 'maria.akhtar@example.com', '+923336789012'),

('Kamran', 'Javed', 'kamran.javed@example.com', '+923347890123'),

('Ayesha', 'Aslam', 'ayesha.aslam@example.com', '+923358901234'),

('Tariq', 'Riaz', 'tariq.riaz@example.com', '+923369012345'),

('Nazia', 'Iqbal', 'nazia.iqbal@example.com', '+923379123456'),

('Yasir', 'Ali', 'yasir.ali@example.com', '+923389234567'),

('Saima', 'Hassan', 'saima.hassan@example.com', '+923399345678'),

('Jamal', 'Ansari', 'jamal.ansari@example.com', '+923409456789'),

('Noman', 'Qureshi', 'noman.qureshi@example.com', '+923419567890'),

('Hina', 'Latif', 'hina.latif@example.com', '+923429678901'),

('Shahzad', 'Mirza', 'shahzad.mirza@example.com', '+923439789012'),

('Asif', 'Bhatti', 'asif.bhatti@example.com', '+923449890123'),

('Rubina', 'Gill', 'rubina.gill@example.com', '+923459901234'),

('Kashif', 'Nawaz', 'kashif.nawaz@example.com', '+923460012345'),

('Sumaira', 'Chaudhry', 'sumaira.chaudhry@example.com', '+923471234567'),

('Waqas', 'Mansoor', 'waqas.mansoor@example.com', '+923482345678'),

('Farah', 'Tanveer', 'farah.tanveer@example.com', '+923493456789'),

('Omar', 'Naeem', 'omar.naeem@example.com', '+923504567890'),

('Saira', 'Adeel', 'saira.adeel@example.com', '+923515678901'),

('Anwar', 'Shafi', 'anwar.shafi@example.com', '+923526789012'),

('Hamza', 'Shah', 'hamza.shah@example.com', '+923537890123'),

('Nadia', 'Rashid', 'nadia.rashid@example.com', '+923548901234'),

('Raza', 'Kamal', 'raza.kamal@example.com', '+923559012345'),

('Zubair', 'Nasir', 'zubair.nasir@example.com', '+923560123456');

-- Inserting data into Properties

INSERT INTO Properties (Address, City, State, PostalCode, Country, OwnerID)

VALUES

('123 Gulberg St', 'Lahore', 'Punjab', '54000', 'Pakistan', 1),

('45 DHA Phase 5', 'Karachi', 'Sindh', '75500', 'Pakistan', 2),

('78 Mall Road', 'Islamabad', 'Capital Territory', '44000', 'Pakistan', 1),

('10 Bahria Town', 'Rawalpindi', 'Punjab', '46000', 'Pakistan', 3),

('12 Gulshan Block', 'Lahore', 'Punjab', '54000', 'Pakistan', 4),

('15 Clifton', 'Karachi', 'Sindh', '75500', 'Pakistan', 5),

('23 Blue Area', 'Islamabad', 'Capital Territory', '44000', 'Pakistan', 1),

('18 G-10', 'Islamabad', 'Capital Territory', '44000', 'Pakistan', 2),

('9 Faisal Town', 'Lahore', 'Punjab', '54000', 'Pakistan', 3),

('22 Cantt', 'Lahore', 'Punjab', '54000', 'Pakistan', 4),

('40 DHA Phase 6', 'Karachi', 'Sindh', '75500', 'Pakistan', 6),

('30 F-7', 'Islamabad', 'Capital Territory', '44000', 'Pakistan', 7),

('55 Gulistan-e-Johar', 'Karachi', 'Sindh', '75290', 'Pakistan', 8),

('88 Main Boulevard', 'Lahore', 'Punjab', '54000', 'Pakistan', 9),

('66 PWD Colony', 'Islamabad', 'Capital Territory', '44000', 'Pakistan', 10),

('14 Johar Town', 'Lahore', 'Punjab', '54000', 'Pakistan', 11),

('29 Model Town', 'Lahore', 'Punjab', '54000', 'Pakistan', 12),

('56 F-10', 'Islamabad', 'Capital Territory', '44000', 'Pakistan', 13),

('17 Clifton Block 9', 'Karachi', 'Sindh', '75500', 'Pakistan', 14),

('67 I-8', 'Islamabad', 'Capital Territory', '44000', 'Pakistan', 15),

('100 DHA Phase 2', 'Islamabad', 'Capital Territory', '44000', 'Pakistan', 16),

('77 DHA Phase 3', 'Lahore', 'Punjab', '54000', 'Pakistan', 17),

('81 Askari 10', 'Rawalpindi', 'Punjab', '46000', 'Pakistan', 18),

('95 Gulshan-e-Iqbal', 'Karachi', 'Sindh', '75290', 'Pakistan', 19),

('38 DHA Phase 5', 'Karachi', 'Sindh', '75500', 'Pakistan', 20),

('48 Gulshan Block 7', 'Lahore', 'Punjab', '54000', 'Pakistan', 21),

('62 Blue Area', 'Islamabad', 'Capital Territory', '44000', 'Pakistan', 22),

('21 Garden Town', 'Lahore', 'Punjab', '54000', 'Pakistan', 23),

('33 Model Town Ext', 'Lahore', 'Punjab', '54000', 'Pakistan', 24),

('44 Gulberg 3', 'Lahore', 'Punjab', '54000', 'Pakistan', 25);

-- Inserting data into Clients

INSERT INTO Clients (FirstName, LastName, Email, Phone)

VALUES

('Ali', 'Raza', 'ali.raza@example.com', '+923111234567'),

('Ayesha', 'Saeed', 'ayesha.saeed@example.com', '+923117654321'),

('Asim', 'Bashir', 'asim.bashir@example.com', '+923212345678'),

('Nadia', 'Iqbal', 'nadia.iqbal@example.com', '+923223456789'),

('Hassan', 'Sadiq', 'hassan.sadiq@example.com', '+923234567890'),

('Sara', 'Rashid', 'sara.rashid@example.com', '+923245678901'),

('Musa', 'Khalid', 'musa.khalid@example.com', '+923256789012'),

('Amna', 'Zahid', 'amna.zahid@example.com', '+923267890123'),

('Usman', 'Farooq', 'usman.farooq@example.com', '+923278901234'),

('Fahad', 'Ali', 'fahad.ali@example.com', '+923289012345'),

('Zara', 'Mehmood', 'zara.mehmood@example.com', '+923299123456'),

('Talha', 'Asif', 'talha.asif@example.com', '+923301234567'),

('Nida', 'Riaz', 'nida.riaz@example.com', '+923312345678'),

('Haris', 'Mumtaz', 'haris.mumtaz@example.com', '+923323456789'),

('Tuba', 'Jamal', 'tuba.jamal@example.com', '+923334567890'),

('Sohail', 'Awan', 'sohail.awan@example.com', '+923345678901'),

('Arfa', 'Latif', 'arfa.latif@example.com', '+923356789012'),

('Bilal', 'Iqbal', 'bilal.iqbal@example.com', '+923367890123'),

('Mariam', 'Nadeem', 'mariam.nadeem@example.com', '+923378901234'),

('Farhan', 'Raza', 'farhan.raza@example.com', '+923389012345'),

('Kiran', 'Zafar', 'kiran.zafar@example.com', '+923390123456'),

('Rameez', 'Akram', 'rameez.akram@example.com', '+923401234567'),

('Uzma', 'Siddiqui', 'uzma.siddiqui@example.com', '+923412345678'),

('Danish', 'Babar', 'danish.babar@example.com', '+923423456789'),

('Sadia', 'Naseer', 'sadia.naseer@example.com', '+923434567890'),

('Noman', 'Hussain', 'noman.hussain@example.com', '+923445678901'),

('Erum', 'Ahmed', 'erum.ahmed@example.com', '+923456789012'),

('Taimoor', 'Javed', 'taimoor.javed@example.com', '+923467890123'),

('Mehwish', 'Sultan', 'mehwish.sultan@example.com', '+923478901234'),

('Saad', 'Ali', 'saad.ali@example.com', '+923489012345');

-- Inserting data into Listings

INSERT INTO Listings (PropertyID, ListingType, Price, ListingDate, Status)

VALUES

(1, 'Sale', 15000000.00, '2024-01-10', 'Available'),

(2, 'Rent', 50000.00, '2024-01-15', 'Available'),

(3, 'Sale', 20000000.00, '2024-02-20', 'Sold'),

(2, 'Rent', 55000.00, '2024-03-01', 'Rented'),

(4, 'Sale', 25000000.00, '2024-03-15', 'Available'),

(5, 'Rent', 60000.00, '2024-04-01', 'Available'),

(6, 'Sale', 30000000.00, '2024-04-10', 'Available'),

(7, 'Rent', 75000.00, '2024-04-20', 'Rented'),

(8, 'Sale', 35000000.00, '2024-05-05', 'Sold'),

(9, 'Rent', 45000.00, '2024-05-10', 'Available'),

(10, 'Sale', 40000000.00, '2024-05-15', 'Available'),

(11, 'Rent', 70000.00, '2024-05-20', 'Rented'),

(12, 'Sale', 55000000.00, '2024-06-01', 'Available'),

(13, 'Rent', 80000.00, '2024-06-10', 'Rented'),

(14, 'Sale', 15000000.00, '2024-06-15', 'Available'),

(15, 'Rent', 65000.00, '2024-06-20', 'Available'),

(16, 'Sale', 25000000.00, '2024-07-01', 'Available'),

(17, 'Rent', 85000.00, '2024-07-05', 'Available'),

(18, 'Sale', 45000000.00, '2024-07-10', 'Available'),

(19, 'Rent', 75000.00, '2024-07-15', 'Available'),

(20, 'Sale', 55000000.00, '2024-07-20', 'Sold'),

(21, 'Rent', 50000.00, '2024-07-25', 'Available'),

(22, 'Sale', 60000000.00, '2024-08-01', 'Available'),

(23, 'Rent', 90000.00, '2024-08-05', 'Available'),

(24, 'Sale', 30000000.00, '2024-08-10', 'Available'),

(25, 'Rent', 65000.00, '2024-08-15', 'Available'),

(26, 'Sale', 45000000.00, '2024-08-20', 'Available'),

(27, 'Rent', 55000.00, '2024-08-25', 'Available'),

(28, 'Sale', 50000000.00, '2024-09-01', 'Available'),

(29, 'Rent', 75000.00, '2024-09-05', 'Available'),

(30, 'Sale', 65000000.00, '2024-09-10', 'Available');

-- Inserting data into Sales

INSERT INTO Sales (ListingID, ClientID, SaleDate, SalePrice)

VALUES

(3, 1, '2024-04-10', 19500000.00),

(8, 3, '2024-05-25', 34000000.00),

(10, 5, '2024-06-10', 39500000.00),

(14, 6, '2024-07-05', 15000000.00),

(18, 7, '2024-07-15', 45000000.00),

(20, 8, '2024-07-25', 55000000.00),

(22, 9, '2024-08-05', 60000000.00),

(24, 10, '2024-08-15', 30000000.00),

(26, 11, '2024-08-25', 45000000.00),

(28, 12, '2024-09-05', 50000000.00);

-- Inserting data into Rents

INSERT INTO Rents (ListingID, ClientID, StartDate, EndDate, RentAmount)

VALUES

(4, 2, '2024-03-15', '2025-03-15', 55000.00),

(7, 4, '2024-05-01', '2024-12-31', 75000.00),

(11, 5, '2024-05-20', '2025-05-20', 70000.00),

(13, 6, '2024-06-10', '2025-06-10', 80000.00),

(15, 7, '2024-06-20', '2025-06-20', 65000.00),

(17, 8, '2024-07-05', '2025-07-05', 85000.00),

(19, 9, '2024-07-15', '2025-07-15', 75000.00),

(21, 10, '2024-07-25', '2025-07-25', 50000.00),

(23, 11, '2024-08-05', '2025-08-05', 90000.00),

(25, 12, '2024-08-15', '2025-08-15', 65000.00),

(27, 13, '2024-08-25', '2025-08-25', 55000.00),

(29, 14, '2024-09-05', '2025-09-05', 75000.00),

(9, 15, '2024-09-10', '2025-09-10', 45000.00),

(6, 16, '2024-09-15', '2025-09-15', 70000.00),

(10, 17, '2024-09-20', '2025-09-20', 80000.00),

(28, 18, '2024-09-25', '2025-09-25', 75000.00),

(3, 19, '2024-09-30', '2025-09-30', 65000.00),

(12, 20, '2024-10-05', '2025-10-05', 60000.00),

(5, 21, '2024-10-10', '2025-10-10', 55000.00),

(2, 22, '2024-10-15', '2025-10-15', 50000.00);

-- Inserting data into Payments

INSERT INTO Payments (ClientID, RentID, Amount, PaymentDate, PaymentMethod)

VALUES

(2, 1, 55000.00, '2024-04-01', 'Bank Transfer'),

(2, 1, 55000.00, '2024-05-01', 'Cash'),

(4, 2, 75000.00, '2024-06-01', 'Credit Card'),

(4, 2, 75000.00, '2024-07-01', 'Bank Transfer'),

(5, 3, 70000.00, '2024-08-01', 'Cash'),

(5, 3, 70000.00, '2024-09-01', 'Credit Card'),

(6, 4, 80000.00, '2024-07-10', 'Bank Transfer'),

(6, 4, 80000.00, '2024-08-10', 'Cash'),

(7, 5, 65000.00, '2024-07-20', 'Bank Transfer'),

(7, 5, 65000.00, '2024-08-20', 'Credit Card'),

(8, 6, 85000.00, '2024-08-05', 'Bank Transfer'),

(8, 6, 85000.00, '2024-09-05', 'Cash'),

(9, 7, 75000.00, '2024-08-15', 'Credit Card'),

(9, 7, 75000.00, '2024-09-15', 'Bank Transfer'),

(10, 8, 50000.00, '2024-08-25', 'Cash'),

(10, 8, 50000.00, '2024-09-25', 'Credit Card'),

(11, 9, 90000.00, '2024-09-05', 'Bank Transfer'),

(11, 9, 90000.00, '2024-10-05', 'Cash'),

(12, 10, 65000.00, '2024-09-15', 'Credit Card'),

(12, 10, 65000.00, '2024-10-15', 'Bank Transfer'),

(13, 11, 55000.00, '2024-09-25', 'Cash'),

(13, 11, 55000.00, '2024-10-25', 'Credit Card'),

(14, 12, 75000.00, '2024-10-05', 'Bank Transfer'),

(14, 12, 75000.00, '2024-11-05', 'Cash'),

(15, 13, 45000.00, '2024-10-10', 'Credit Card'),

(15, 13, 45000.00, '2024-11-10', 'Bank Transfer'),

(16, 14, 70000.00, '2024-10-15', 'Cash'),

(16, 14, 70000.00, '2024-11-15', 'Credit Card'),

(17, 15, 80000.00, '2024-10-20', 'Bank Transfer'),

(17, 15, 80000.00, '2024-11-20', 'Cash'),

(18, 16, 75000.00, '2024-10-25', 'Credit Card'),

(18, 16, 75000.00, '2024-11-25', 'Bank Transfer');

-- Inserting 30 rows into MaintenanceRequests table

INSERT INTO MaintenanceRequests (PropertyID, ClientID, Description, RequestDate, Status)

VALUES

(1, 1, 'Water leakage in the kitchen', '2024-01-10', 'Pending'),

(2, 2, 'Broken window in the living room', '2024-01-12', 'In Progress'),

(3, 3, 'Electrical wiring issue in the bedroom', '2024-01-15', 'Completed'),

(4, 4, 'Pest control needed in the basement', '2024-01-20', 'Pending'),

(5, 5, 'Plumbing issue in the bathroom', '2024-01-25', 'In Progress'),

(6, 6, 'Air conditioning not working', '2024-01-30', 'Completed'),

(7, 7, 'Heater malfunctioning in the winter', '2024-02-05', 'Pending'),

(8, 8, 'Roof leakage during rain', '2024-02-10', 'In Progress'),

(9, 9, 'Paint peeling in the hallway', '2024-02-15', 'Completed'),

(10, 10, 'Door lock broken', '2024-02-20', 'Pending'),

(11, 11, 'Clogged drainage system', '2024-02-25', 'In Progress'),

(12, 12, 'Gas leakage detected in the kitchen', '2024-03-01', 'Completed'),

(13, 13, 'Garage door not functioning', '2024-03-05', 'Pending'),

(14, 14, 'Water heater issue', '2024-03-10', 'In Progress'),

(15, 15, 'Faucet leak in the bathroom', '2024-03-15', 'Completed'),

(16, 16, 'Mold growth in the bathroom', '2024-03-20', 'Pending'),

(17, 17, 'Wall crack in the living room', '2024-03-25', 'In Progress'),

(18, 18, 'Floor tiles are coming off', '2024-03-30', 'Completed'),

(19, 19, 'Window jam in the bedroom', '2024-04-05', 'Pending'),

(20, 20, 'Noise issue with HVAC system', '2024-04-10', 'In Progress'),

(21, 21, 'Bathroom exhaust fan not working', '2024-04-15', 'Completed'),

(22, 22, 'Gate remote control malfunctioning', '2024-04-20', 'Pending'),

(23, 23, 'Leaking faucet in the kitchen', '2024-04-25', 'In Progress'),

(24, 24, 'Power outage in certain areas', '2024-04-30', 'Completed'),

(25, 25, 'Sliding door stuck', '2024-05-05', 'Pending'),

(26, 26, 'Water pump not working', '2024-05-10', 'In Progress'),

(27, 27, 'Security camera offline', '2024-05-15', 'Completed'),

(28, 28, 'Garden sprinkler malfunctioning', '2024-05-20', 'Pending'),

(29, 29, 'Basement flooding issue', '2024-05-25', 'In Progress'),

(30, 30, 'Insulation problem in the attic', '2024-05-30', 'Completed');

**JOINS**

**-- 1. List Properties with Their Landlords**

SELECT

Properties.PropertyID,

Properties.Address,

Properties.City,

Properties.State,

Properties.PostalCode,

Properties.Country,

Landlords.FirstName AS LandlordFirstName,

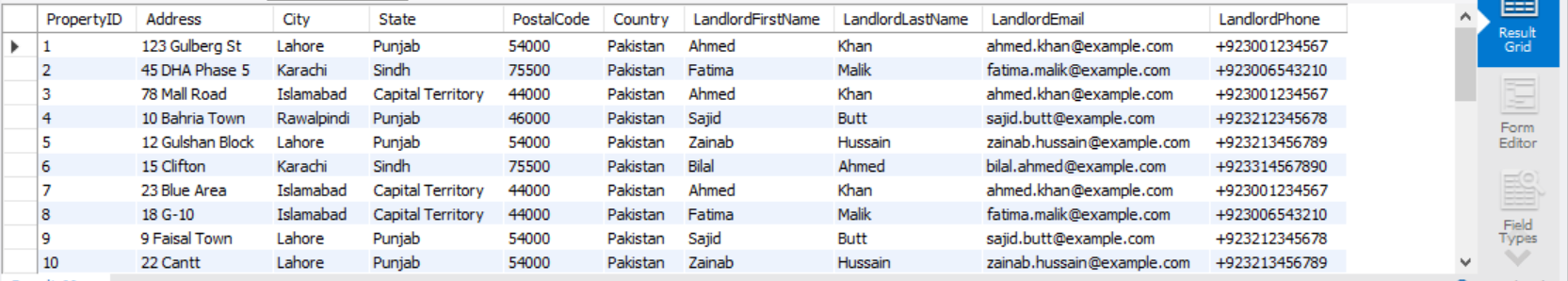
Landlords.LastName AS LandlordLastName,

Landlords.Email AS LandlordEmail,

Landlords.Phone AS LandlordPhone

FROM Properties

JOIN Landlords ON Properties.OwnerID = Landlords.LandlordID;



**-- 2. List All Listings with Their Property Details and Status**

SELECT

Listings.ListingID,

Properties.Address,

Properties.City,

Properties.State,

Properties.PostalCode,

Listings.ListingType,

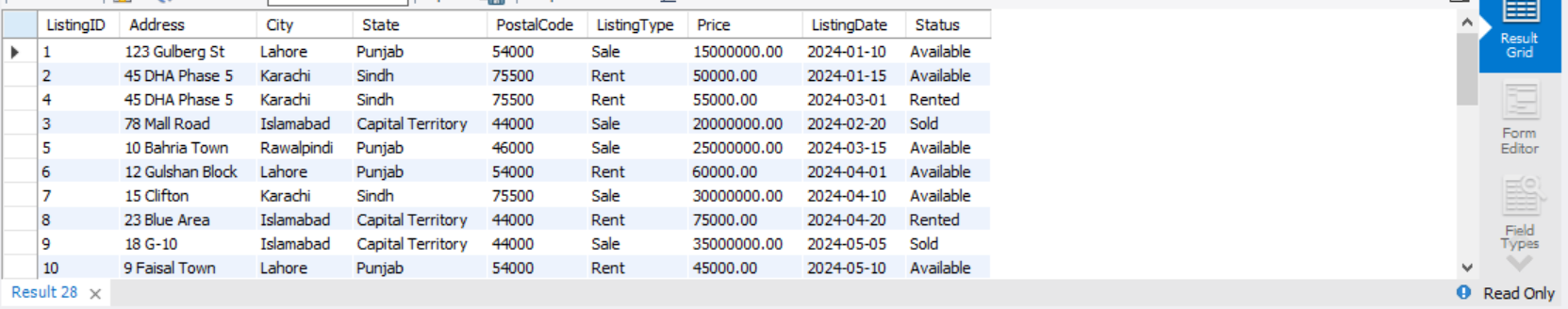
Listings.Price,

Listings.ListingDate,

Listings.Status

FROM Listings

JOIN Properties ON Listings.PropertyID = Properties.PropertyID;



**-- 3. List of Clients Who Purchased Properties**

SELECT

Sales.SaleID,

Clients.FirstName AS ClientFirstName,

Clients.LastName AS ClientLastName,

Clients.Email AS ClientEmail,

Properties.Address,

Properties.City,

Properties.State,

Properties.PostalCode,

Sales.SaleDate,

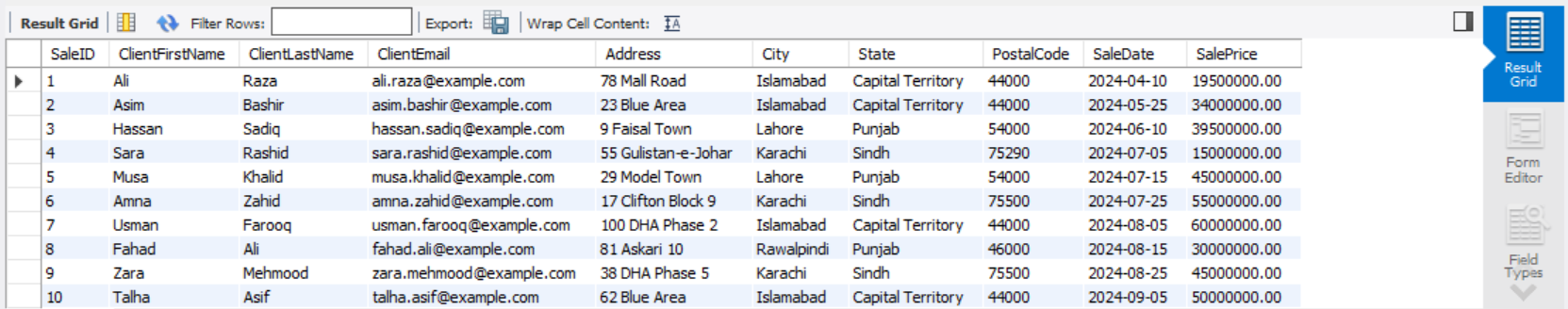
Sales.SalePrice

FROM Sales

JOIN Clients ON Sales.ClientID = Clients.ClientID

JOIN Listings ON Sales.ListingID = Listings.ListingID

JOIN Properties ON Listings.PropertyID = Properties.PropertyID;



**-- 4. List of Clients Who Are Renting Properties**

SELECT

Rents.RentID,

Clients.FirstName AS ClientFirstName,

Clients.LastName AS ClientLastName,

Clients.Email AS ClientEmail,

Properties.Address,

Properties.City,

Properties.State,

Properties.PostalCode,

Rents.StartDate,

Rents.EndDate,

Rents.RentAmount

FROM Rents

JOIN Clients ON Rents.ClientID = Clients.ClientID

JOIN Listings ON Rents.ListingID = Listings.ListingID

JOIN Properties ON Listings.PropertyID = Properties.PropertyID;



**-- 5. List of Payments Made by Clients**

SELECT

Payments.PaymentID,

Clients.FirstName AS ClientFirstName,

Clients.LastName AS ClientLastName,

Payments.Amount,

Payments.PaymentDate,

Payments.PaymentMethod,

Properties.Address AS PropertyAddress,

Rents.StartDate,

Rents.EndDate

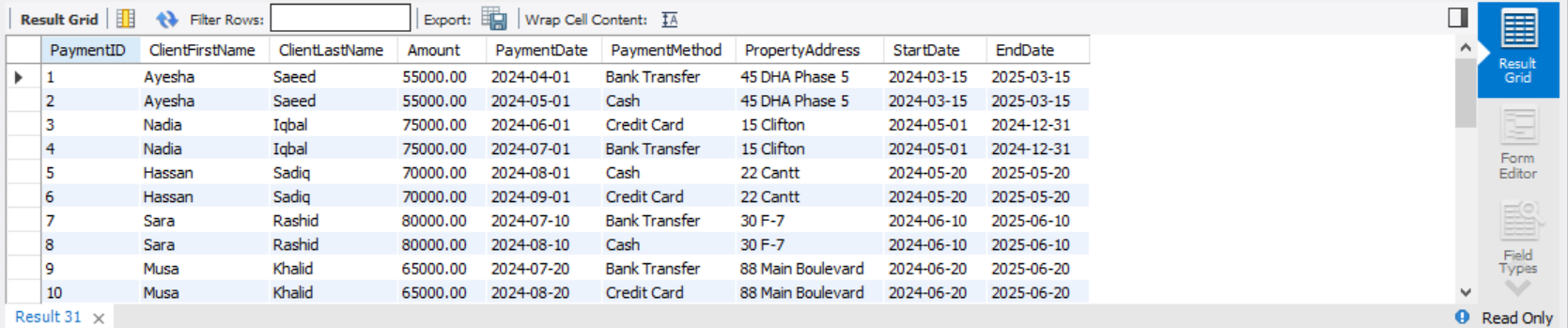
FROM Payments

JOIN Clients ON Payments.ClientID = Clients.ClientID

JOIN Rents ON Payments.RentID = Rents.RentID

JOIN Listings ON Rents.ListingID = Listings.ListingID

JOIN Properties ON Listings.PropertyID = Properties.PropertyID;



**-- 6. List of Maintenance Requests by Clients**

SELECT

DISTINCT MaintenanceRequests.RequestID,

Clients.FirstName AS ClientFirstName,

Clients.LastName AS ClientLastName,

Properties.Address,

MaintenanceRequests.Description,

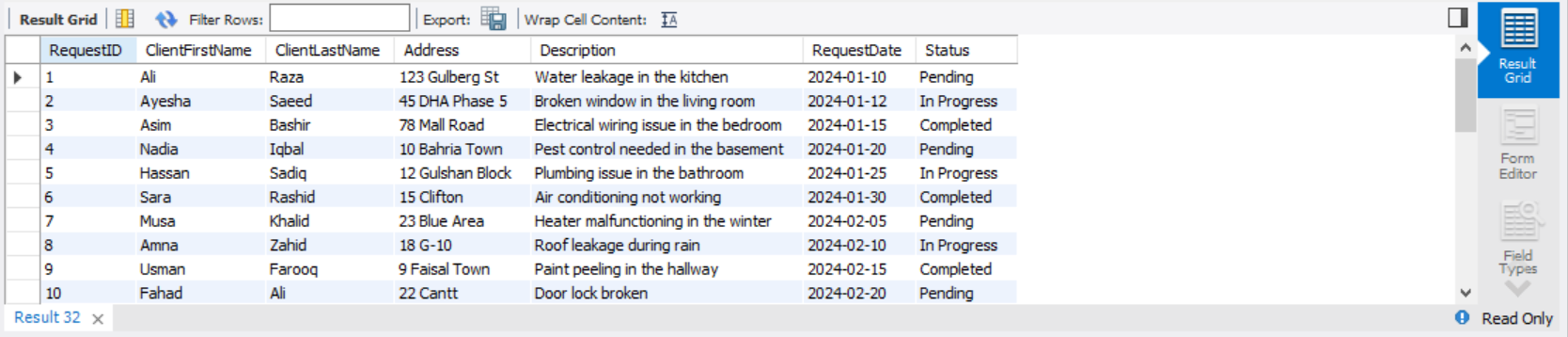
MaintenanceRequests.RequestDate,

MaintenanceRequests.Status

FROM MaintenanceRequests

JOIN Clients ON MaintenanceRequests.ClientID = Clients.ClientID

JOIN Properties ON MaintenanceRequests.PropertyID = Properties.PropertyID;



**-- 7. List Properties That Are Currently Rented Out**

SELECT

Properties.PropertyID,

Properties.Address,

Properties.City,

Properties.State,

Properties.PostalCode,

Clients.FirstName AS ClientFirstName,

Clients.LastName AS ClientLastName,

Rents.StartDate,

Rents.EndDate,

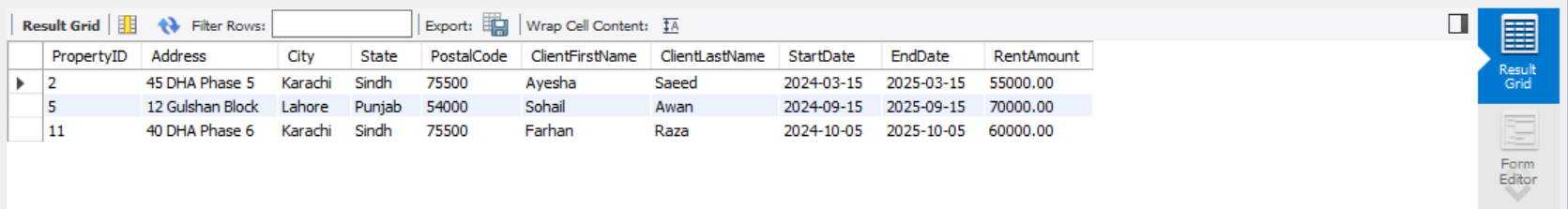
Rents.RentAmount

FROM Properties

JOIN Listings ON Properties.PropertyID = Listings.PropertyID

JOIN Rents ON Listings.ListingID = Rents.ListingID

JOIN Clients ON Rents.ClientID = Clients.ClientID

WHERE Listings.ListingType = 'Rent' AND Listings.Status = 'Rented';

**-- 8. List of Available Properties for Sale**

SELECT

Properties.PropertyID,

Properties.Address,

Properties.City,

Properties.State,

Properties.PostalCode,

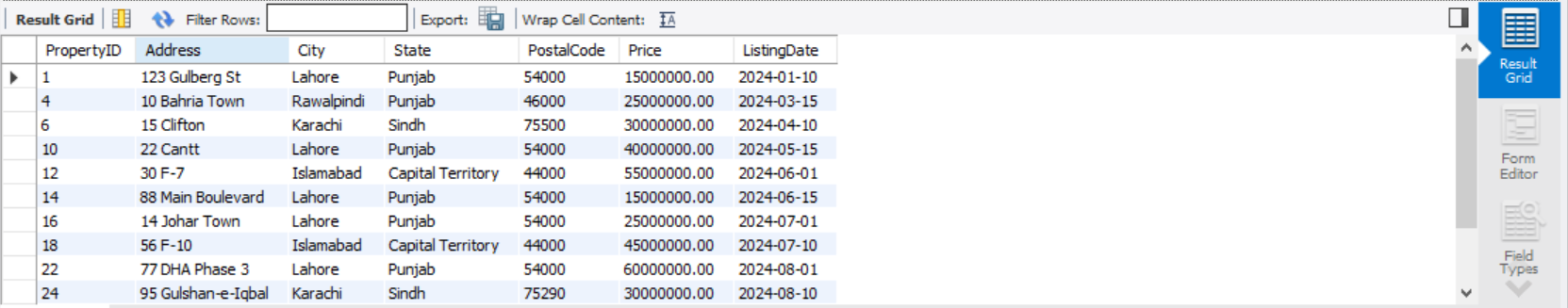
Listings.Price,

Listings.ListingDate

FROM Properties

JOIN Listings ON Properties.PropertyID = Listings.PropertyID

WHERE Listings.ListingType = 'Sale' AND Listings.Status = 'Available';



**-- 9. List All Properties With Their Rental Status**

SELECT

Properties.PropertyID,

Properties.Address,

Properties.City,

Properties.State,

Properties.PostalCode,

Listings.Status AS RentalStatus

FROM Properties

JOIN Listings ON Properties.PropertyID = Listings.PropertyID

WHERE Listings.ListingType = 'Rent';



**-- 10. List Properties With Pending Maintenance Requests**

SELECT

Properties.PropertyID,

Properties.Address,

Properties.City,

Properties.State,

Properties.PostalCode,

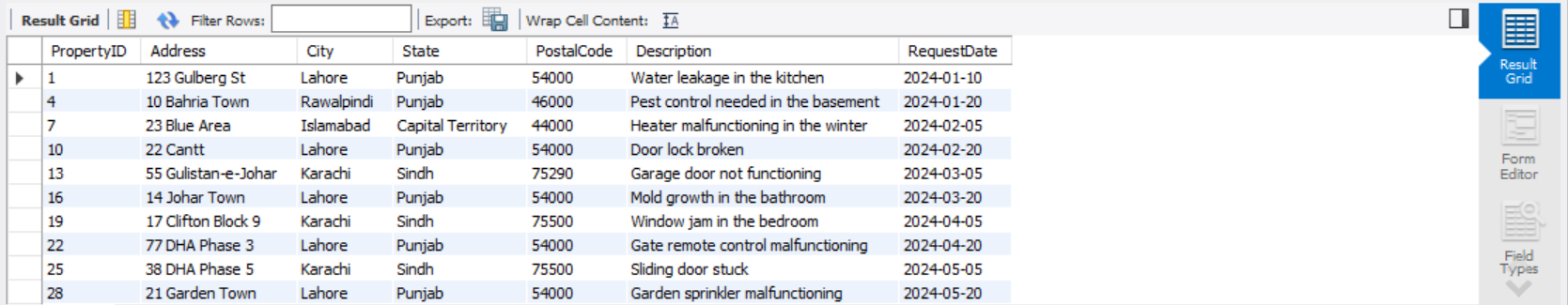
MaintenanceRequests.Description,

MaintenanceRequests.RequestDate

FROM Properties

JOIN MaintenanceRequests ON Properties.PropertyID = MaintenanceRequests.PropertyID

WHERE MaintenanceRequests.Status = 'Pending';



-- Nested Queries

-- 1 Find the clients who have made payments using multiple methods:

SELECT c.ClientID, c.FirstName, c.LastName

FROM Clients c

WHERE EXISTS (

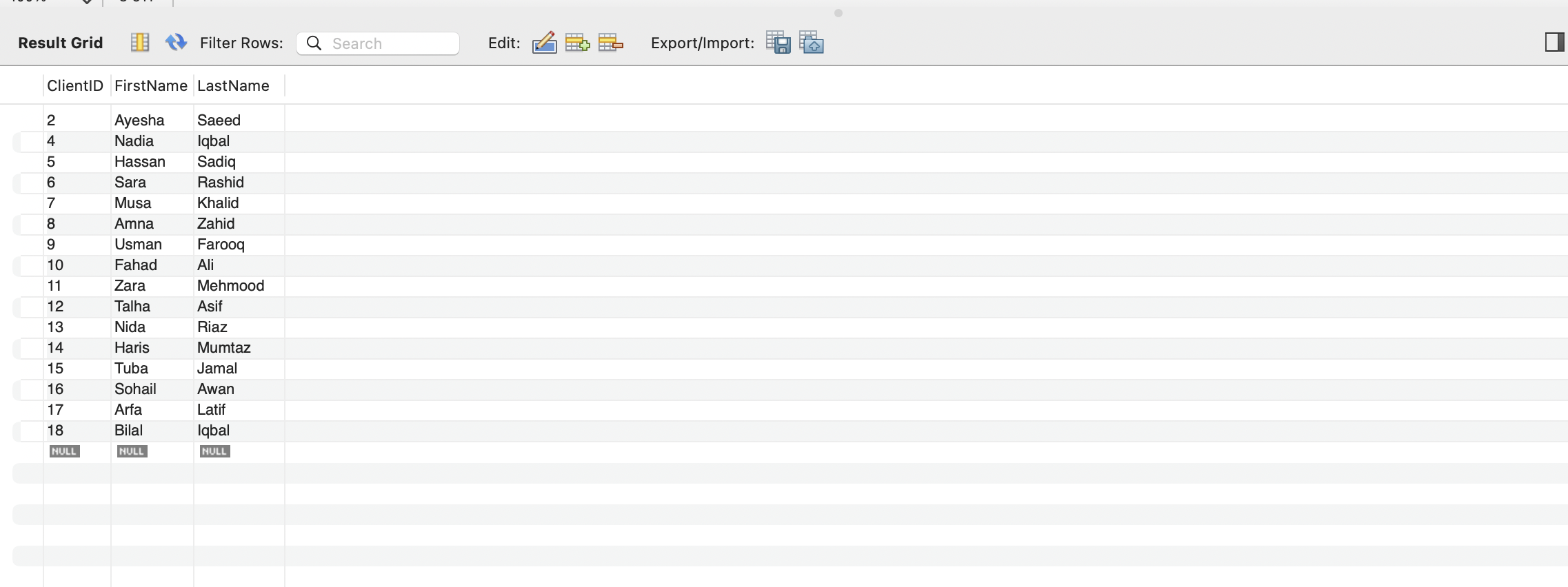
SELECT \*

FROM Payments p1

JOIN Payments p2 ON p1.ClientID = p2.ClientID AND p1.PaymentMethod <> p2.PaymentMethod

WHERE c.ClientID = p1.ClientID

);



-- 2 List the landlords who own properties in more than one city:

SELECT l.LandlordID, l.FirstName, l.LastName

FROM Landlords l

WHERE EXISTS (

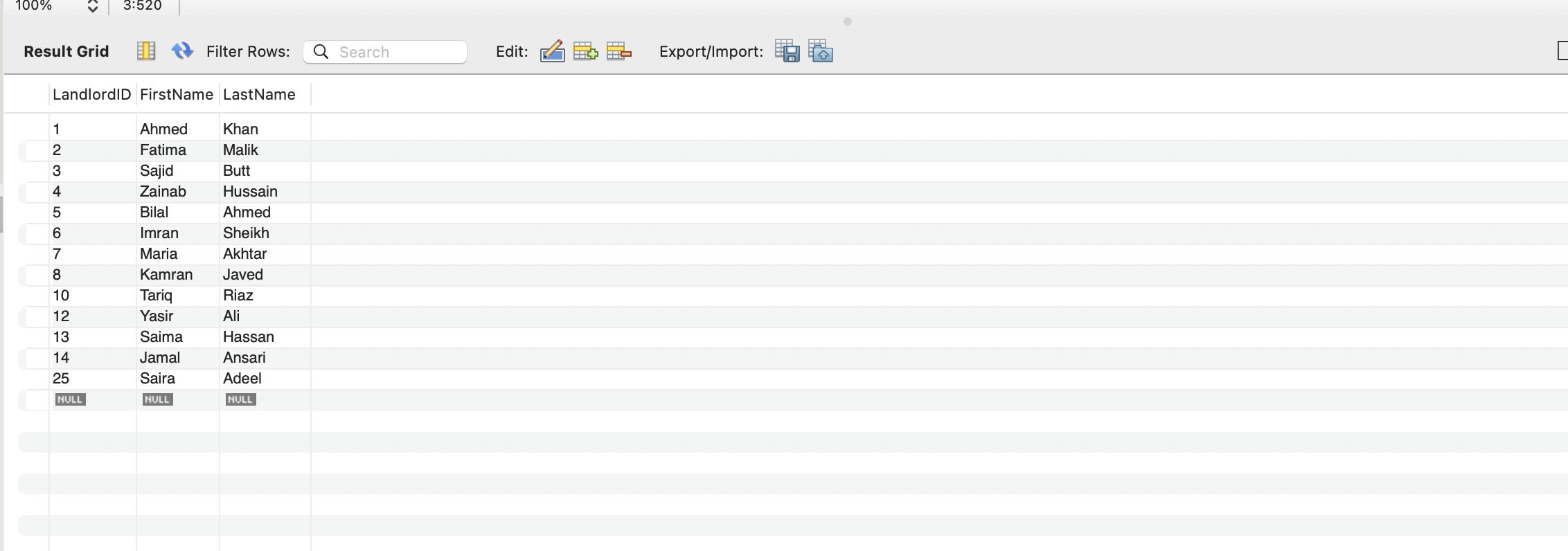
SELECT \*

FROM Properties p1

JOIN Properties p2 ON p1.OwnerID = p2.OwnerID AND p1.City <> p2.City

WHERE l.LandlordID = p1.OwnerID

);



-- 3 Find properties that are either for sale or rent and have had maintenance requests:

SELECT p.PropertyID, p.Address

FROM Properties p

WHERE EXISTS (

SELECT \*

FROM Listings l

WHERE p.PropertyID = l.PropertyID AND (l.Status = 'Available' OR l.Status = 'Rented')

) AND EXISTS (

SELECT 1

FROM MaintenanceRequests mr

WHERE p.PropertyID = mr.PropertyID

);



-- 4 List clients who have purchased a property and have a pending maintenance request for that property:

SELECT c.ClientID, c.FirstName, c.LastName

FROM Clients c

WHERE EXISTS (

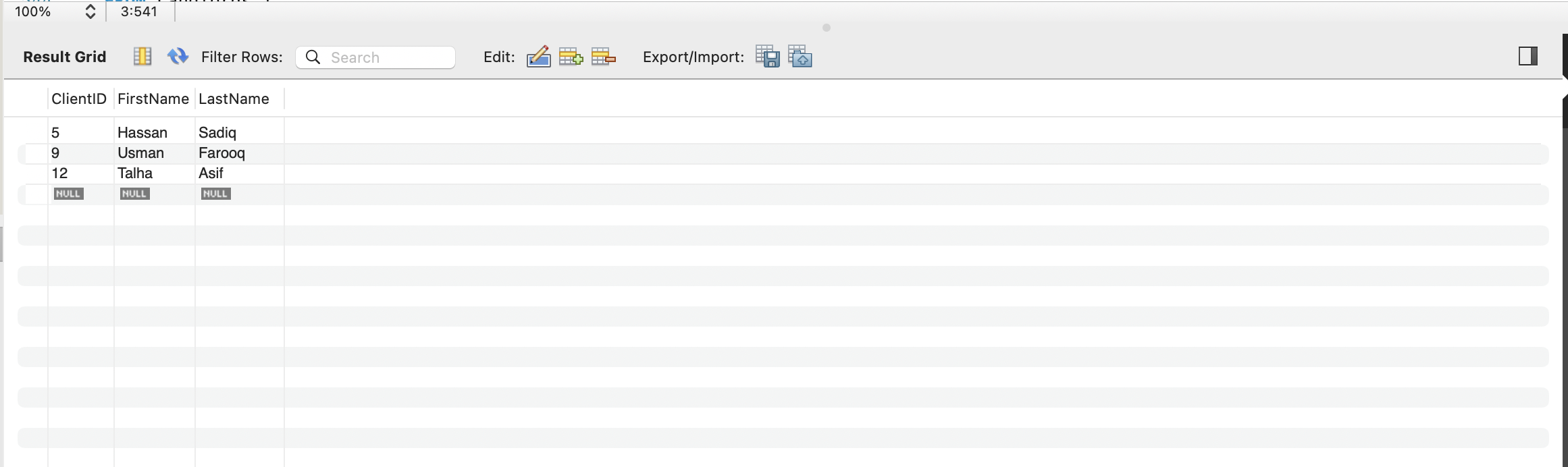
SELECT \*

FROM Sales s

JOIN MaintenanceRequests mr ON s.ListingID = mr.PropertyID

WHERE c.ClientID = s.ClientID AND mr.Status = 'Pending'

);



-- 5 Find landlords whose properties have been rented out and have received payments:

SELECT l.LandlordID, l.FirstName, l.LastName

FROM Landlords l

WHERE EXISTS (

SELECT \*

FROM Properties p

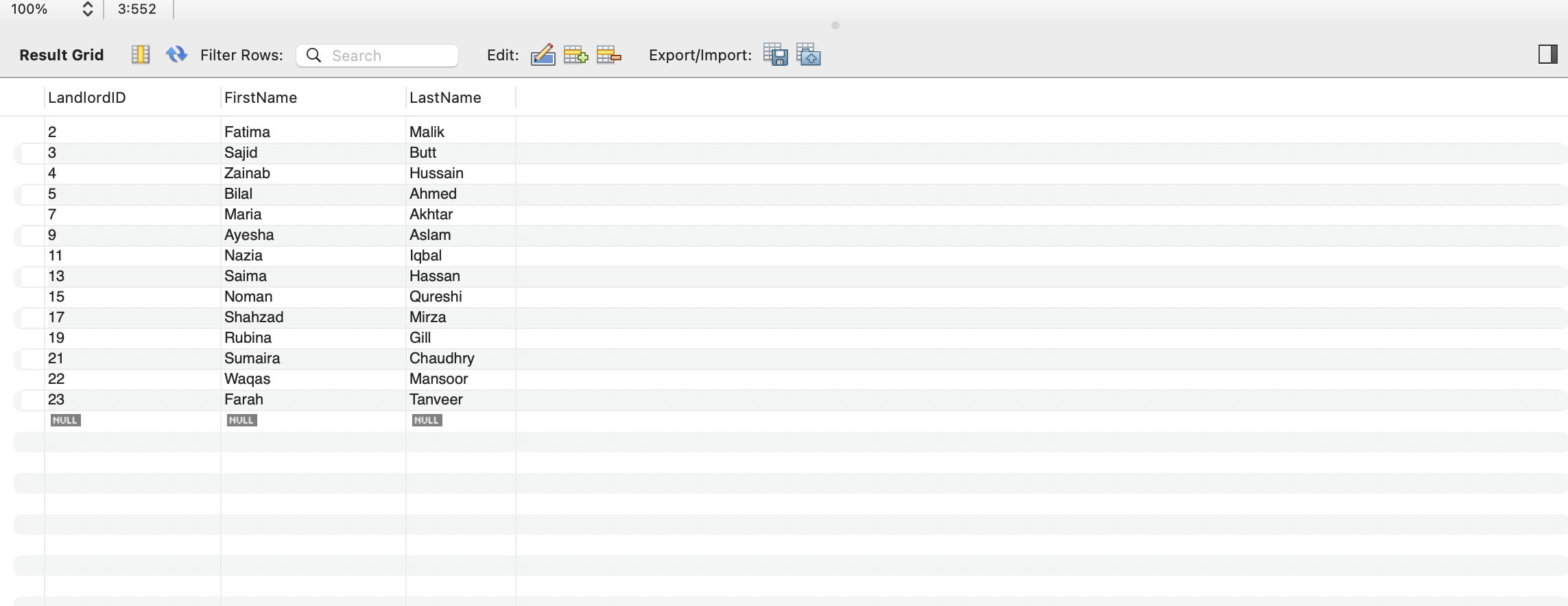
JOIN Listings li ON p.PropertyID = li.PropertyID

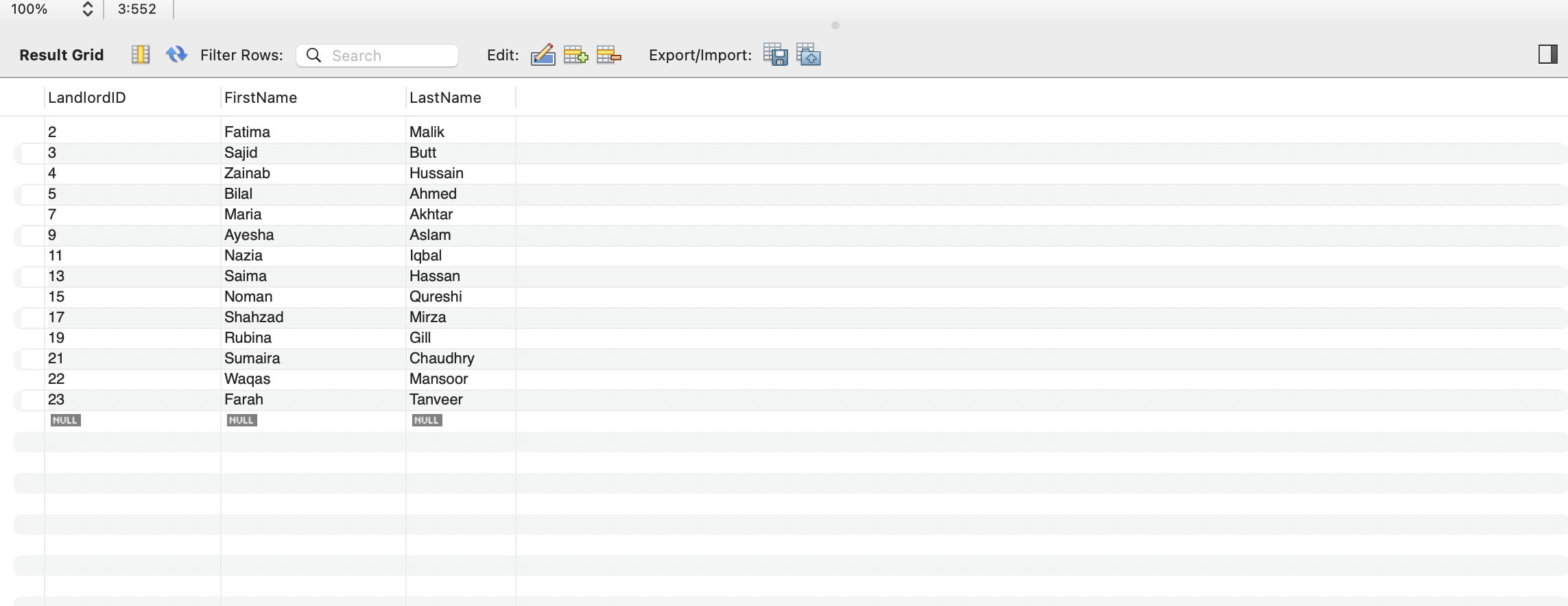
JOIN Rents r ON li.ListingID = r.ListingID

JOIN Payments pay ON r.RentID = pay.RentID

WHERE l.LandlordID = p.OwnerID

);





-- 6 List properties that have been sold to clients who also have rented properties:

SELECT p.PropertyID, p.Address

FROM Properties p

WHERE EXISTS (

SELECT 1

FROM Sales s

WHERE p.PropertyID = s.ListingID

AND EXISTS (

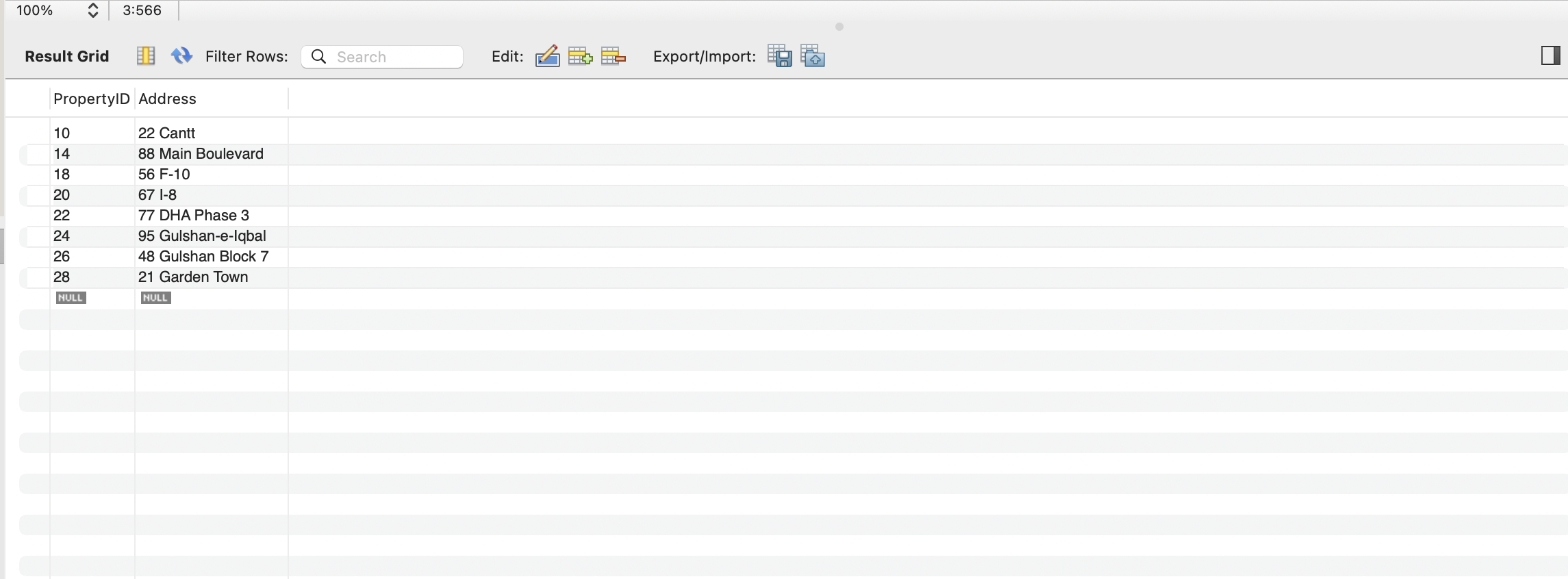
SELECT 1

FROM Rents r

WHERE r.ClientID = s.ClientID

)

);



-- 7 Find the total amount of rent paid by each client for properties they have rented:

SELECT c.ClientID, c.FirstName, c.LastName, (

SELECT SUM(p.Amount)

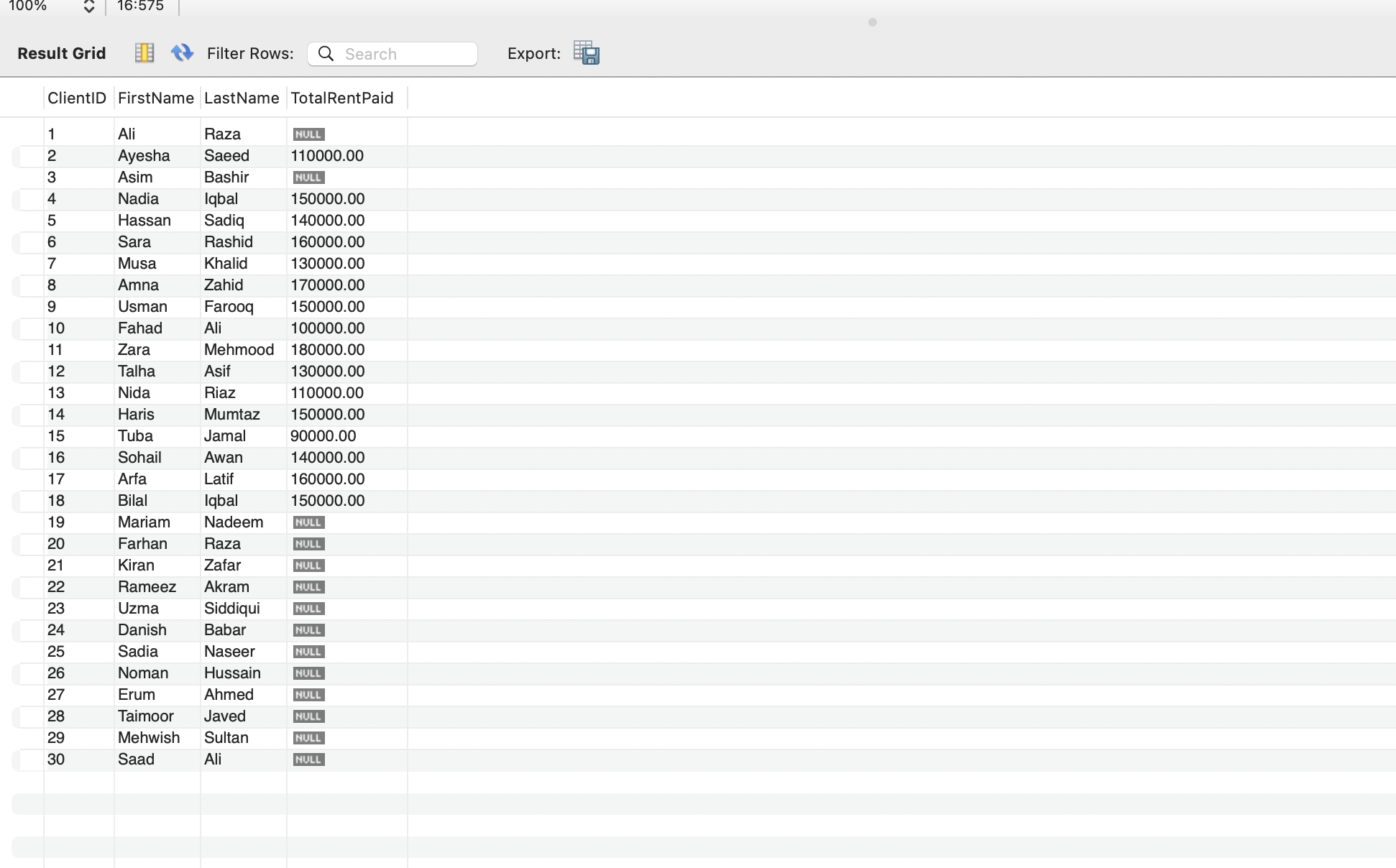
FROM Payments p

JOIN Rents r ON p.RentID = r.RentID

WHERE p.ClientID = c.ClientID

) AS TotalRentPaid

FROM Clients c;



-- 8 Retrieve landlords whose properties have been both sold and rented:

SELECT l.LandlordID, l.FirstName, l.LastName

FROM Landlords l

WHERE EXISTS (

SELECT 1

FROM Properties p

JOIN Listings ls ON p.PropertyID = ls.PropertyID

WHERE l.LandlordID = p.OwnerID AND ls.Status = 'Sold'

) AND EXISTS (

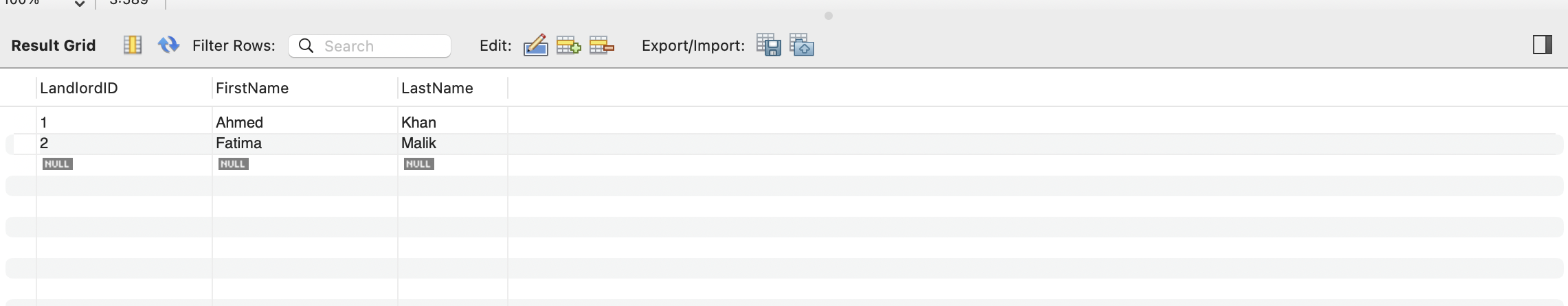
SELECT 1

FROM Properties p

JOIN Listings ls ON p.PropertyID = ls.PropertyID

WHERE l.LandlordID = p.OwnerID AND ls.Status = 'Rented'

);



-- 9 List all maintenance requests for properties that are currently rented:

SELECT mr.RequestID, mr.Description, p.Address

FROM MaintenanceRequests mr

JOIN Properties p ON mr.PropertyID = p.PropertyID

WHERE EXISTS (

SELECT 1

FROM Listings l

JOIN Rents r ON l.ListingID = r.ListingID

WHERE l.PropertyID = p.PropertyID AND l.Status = 'Rented'

);



-- 10 Find properties that have never had a maintenance request:

SELECT p.PropertyID, p.Address

FROM Properties p

WHERE NOT EXISTS (

SELECT 1

FROM MaintenanceRequests mr

WHERE p.PropertyID = mr.PropertyID

);

