Name: Samantha Peters
CSCI250 Database

Domain Description

This is a rental real estate investing company. This company participates in syndication, acquisition, and management of the investment property. The end user of this database is employees within the company.

Functional Requirements

Current Properties: Current properties are rental complexes that are owned by the company. Current properties is defined by the following attributes:

1. Property_id: Each property has its own unique identification number that is a nine digit integer
2. Address: A property must have one address. An address is a composite attribute that is composed of *address_line_one*, *address_line_two*, *city*, *state*, *zip*, and *country*.
3. Num_units: The number of units each property has. A property must have at least one unit.
4. Date: date describes the date of acquisition for the property.
5. Actual_income: The total income the property generated in the prior twelve months.
6. Purchase_price: Price the property was originally purchased at.
7. Num_occupants: Integer that describes the maximum number of occupants who can live at the property. There can only be one occupant per unit.

Offices: There are multiple office locations throughout the United States. Offices is defined by the following attributes:

1. Office_id: Each office location has its own unique identification number that is nine integers long
2. Phone_number: The office phone number.
3. Email: Office email.
4. Address: An office must have one address. An address is a composite attribute that is composed of *address_line_one*, *address_line_two*, *city*, *state*, *zip*, and *country*.

Tenant: The individual residing in the rental complexes. Tenant is defined by the following attributes:

1. Tenant_id: Each tenant has a unique identification number that is nine integers long.
2. Name: Each resident has a name.
3. Address: A property must have one address. An address is a composite attribute that is composed of *address_line_one*, *address_line_two*, *city*, *state*, *zip*, and *country*.
4. Appt_num: Each tenant has exactly one room number.
5. Email: Each tenant needs exactly one email.
6. Phone number: Each tenant needs exactly one phone number.

Prospective Acquisition: The properties that the company is considering investing in or buying. Prospective Acquisition is defined by the following attributes:

1. Prospective_id ; Each prospective acquisition has a unique identification number.
2. Owner_name; Each property has one owner.
3. Owner_email: Each owner has one email address.
4. Owner_phone: Each owner has exactly one phone number.
5. Address: Each property has an address. It is a composite attribute composed of the same attributes as the aforementioned address attributes.
6. Purchare_price: Each property has purchase price.
7. Down_payment: The down payment for the prospective acquisition
8. Initial_payment: The initial payment to open escrow.
9. Inspection_date: Each property will have at least one inspection.

Rent Roll: Is an account/register of the rents and their tenants. This is a table within the Pro Forma for prospective acquisitions that is created by the owner of the prospective property. Rent Roll is defined by the following attributes:

1. Roll_id: Each rent roll is given a unique identification number.
2. Appt_num: The apartment number for the individual apartment.
3. Type: The type of unit and is in the form “1/1”, “2/1”, and so on.
4. SizePerSquare: The unit’s size / square feet.
5. Rent: The units rent per month represented in a dollar amount.

Pro_Forma: A document produced by the owner of the prospective acquisition that contains multiple tables. It provides projections about operating costs, expenses, income, and profits. Pro forma is defined by the following attributes:

1. Proforma_id: Each proforma has a unique identification number.
2. Date: The date the Pro Forma was created. The owner of the prospective acquisition creates and publishes this form.

3. Actual_income: The total income the property generated in the prior twelve months. Represented in a dollar amount.
4. Future_Potential_Income: The total income the property could generate at today's market rents, 100 percent occupancy and taking full advantage of all other income opportunities. Represented in a dollar amount.
5. Actual_Potential_income: The total income the property could generate at today's market rents, 100 percent occupancy and taking full advantage of all other income opportunities. Represented in a dollar amount.

Expenses Per Unit: a table within the Pro Forma that describes the expenses per unit for the prospective acquisition provided by the current owner. Expenses per unit is defined by the following attributes:

1. EPU_id : Each Expenses Per Unit has a unique identification number.
2. Repairs_maintence: The cost of repairs and maintenance per unit in a dollars amount.
3. Utilities: The cost of utilities per unit represented in a dollar amount.
4. Taxes: The cost of real estate taxes per unit represented in a dollar amount.
5. Insurance: The cost of insurance represented in a dollar amount.
6. Reserve: The cost of replacement reserve per unit represented in a dollar amount.
7. Total_expenses: The total cost of expenses for the entire property represented in a dollar amount.

Unit Mix and Rent Schedule: A table within the Pro Forma provided by the owner. Unit mix represents the mix of different apartment types within the apartment complex. Rent Schedule describes the schedule of rents. Unit Mix and Rent Schedule is defined by the following attributes:

1. Mix_id: Each Unit Mix and Rent Schedule has its own unique identification number.
2. Num_units: Represents the total number of units for the property.
3. Type: The type of unit and is in the form "1 bedroom", "2 bedroom", and so on.
4. Square_feet: The size of the unit represented in square feet.
5. Rent: The cost of rent per month represented in a dollar amount per unit.
6. Total_rent: The total rent for the entire property represented in a dollar amount.
7. RentPerSquareFoot: The amount of rent represented by total rent divided by total square foot. ****Note: Realized after creating the DDL and inserting all of my data that when querying an attribute that uses capital letters such as this one and another attribute in Rent_roll, you have to include quotes around the attribute. If I were to redo this project, I would use the snake naming convention instead.

Expenses Seller Pro Forma: Is a table describing the projected expenses that the seller has provided. The data within the table must be verified by the purchasing party. The expenses seller pro form is defined by the following attributes:

1. Expenses_Seller_Proforma_id : Each Expenses Seller Pro Forma has a unique identification number.
2. Repairs_maintence: The cost of repairs and maintenance per unit in a dollars amount.
3. Utilities: The cost of utilities per unit represented in a dollar amount.
4. Taxes: The cost of real estate taxes per unit represented in a dollar amount.
5. Insurance: The cost of insurance represented in a dollar amount.
6. Reserve: The cost of replacement reserve per unit represented in a dollar amount.
7. Total_expenses: The total cost of expenses for the entire property represented in a dollar amount.

Expenses Prior Year Actuals: This is a table that the company does independent research to produce based off information from the seller and is then added to the Pro Forma to compare against the Seller's expenses. Expenses prior year actuals is defined by the following attributes:

1. Prior_id: Each Expenses Prior Year Actuals has a unique identification number.
2. Repairs_maintence: The cost of repairs and maintenance per unit in a dollars amount.
3. Utilities: The cost of utilities per unit represented in a dollar amount.
4. Taxes: The cost of real estate taxes per unit represented in a dollar amount.
5. Insurance: The cost of insurance represented in a dollar amount.
6. Reserve: The cost of replacement reserve per unit represented in a dollar amount.
7. Total_expenses: The total cost of expenses for the entire property represented in a dollar amount.

Projected Expenses: Based off research conducted by the company, the projected expenses are produced and added to the Pro Forma. This is used to determine the value of the property. The project expenses table is defined by the following attributes:

1. Projected_id: Each Projected Expenses has a unique identification number.
2. Repairs_maintence: The cost of repairs and maintenance per unit in a dollars amount.
3. Utilities: The cost of utilities per unit represented in a dollar amount.
4. Taxes: The cost of real estate taxes per unit represented in a dollar amount.
5. Insurance: The cost of insurance represented in a dollar amount.
6. Reserve: The cost of replacement reserve per unit represented in a dollar amount.
7. Total_expenses: The total cost of expenses for the entire property represented in a dollar amount.

Net Operating Income: Take projected total income and subtract projected total expenses to get net operating income. Used to evaluate the value of the prospective acquisition. Net Operating Income table Take projected total income and subtract projected total expenses to get net operating income

1. NOI_ID: Each Net Operating Income table receives a unique identification number.
2. Income_total: The total income of the property given in a dollar amount.

3. Expense_total: The total expense of the property given in a dollar amount.
4. noi_total: is income total subtracted by expense total.

Employee: An employee working full or part time at the company. An employee is defined by the following attributes:

1. Employee_id: Each employee receives a unique nine digit identification number.
2. Team: Each employee must work for exactly one team. Teams include Accounting and Finance Team, Real Estate Team, Law and Taxes Team, Marketing Team, HR Team, Management Team, Administration Team, Syndication Team, Investor Relations Team, Property Management Team, Maintenance Team, and Consultant Team.
3. Role: Each employee must have exactly one role. The role describes their position/responsibilities within their prospective team. An example is “Accountant”, “Attorney”, “Sponser”, and “Maintenance Tech”.
4. Address: A property must have one address. An address is a composite attribute that is composed of *address_line_one*, *address_line_two*, *city*, *state*, *zip*, and *country*.
5. Phone_number: Each employee must have one personal phone number on file.
6. Email: Each employee must have exactly one email address.
7. Salary: Each employee has a salary represented in a dollar amount although it might be null when the employee is first hired.

Lease Agreement: A lease agreement is essentially a contract between the tenant and landlord that the tenant will pay the agreed upon amount to the landlord on the agreed upon periods. A lease agreement is defined by the following attributes:

1. Lease_id; Each lease agreement has a unique identification number that belongs to each tenant and their lease.
2. Date: The date the lease agreement was signed and processed. Each lease agreement must have one.
3. Lease_term: The team of the lease it can be “month-to-month” or “year-to-year”.
4. Rent: The amount represented in dollars that the rent is each month.
5. Rent_due: The date the rent is due each month.
6. Payment_Ins: Payment instructions for the lease. It is a note. It can be null.
7. Fee_date: The date that a late fee would be issued.
8. Fee_amount: The amount represented in dollars that the late fee would be if issued.

Landlord: A landlord is the designated property manager who oversees the day to day responsibilities of the property. A landlord is defined by the following attributes:

1. Landlord_id: Each landlord has a unique identification number that is nine integers long.

Loans: A loan is money lent by lender institutions for acquisitions. A loan is defined by the following attributes:

1. Loan_id: Each loan receives a unique identification number.
2. Name: The name of the bank or lender institution that is issuing the loan.
3. Phone_number: The phone number for the bank or lender institution issuing the loan.
4. Email: The email for the bank or lender institution issuing the loan.
5. Amount: The principal of the loan is represented in dollars.
6. Rate: The interest rate for the loan is represented as a percentage.
7. Payment: The payment amount represented in dollars that is due each team.
8. Term: The term of the loan which can be months or years.
9. Date: The date the loan was issued.

General Manager: An employee of the company who is in charge of overseeing the syndication and acquisition of rental properties. A general manager is defined by the following attributes:

1. General_id: Each general manager has a unique identification number.
2. Name: the general manager's name.
3. phone number: the general manager's name.

Syndication: Real estate investors and other parties pool together resources and capital to purchase real estate investments. Syndication is defined by the following attributes:

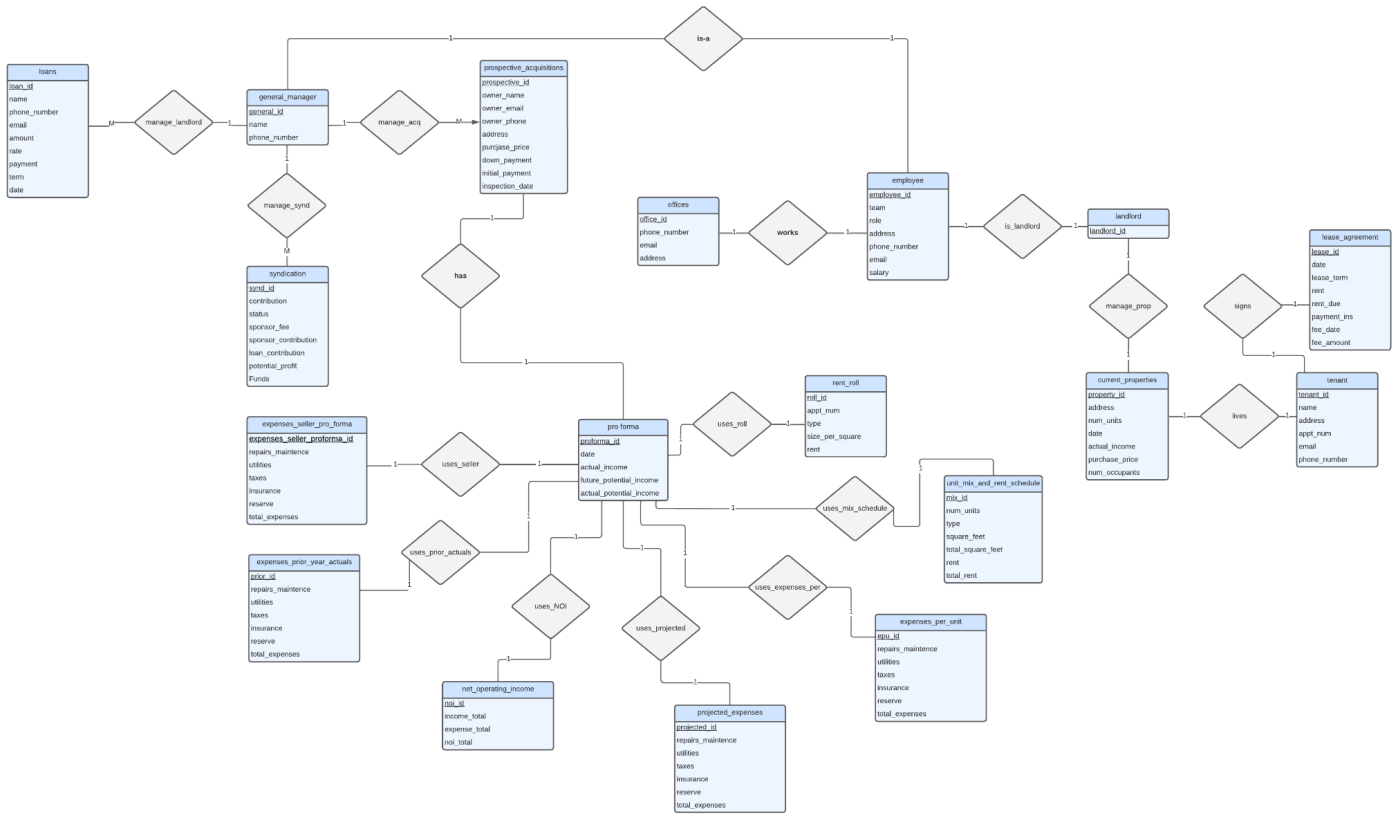
1. Synd_id: A syndication is essentially a deal that involves investors and a sponsor. Each syndication has a unique identification number.
2. Contribution: This describes the total contribution coming from outside sources.
3. Status: This is a note or description that describes the status of syndication.
4. Sponser_fee: The company, in the syndication, is the sponsor and takes 7-25% of the potential profit as a fee for managing the property and overseeing the syndication.
5. Sponsor_contribution: The total amount represented in dollars that the sponsors (or company) has invested into the deal.
6. Loan_contribution: The total amount represented in dollars that the bank/loan has contributed to the deal.
7. Potential_profit: The potential profit for the prospective acquisition after analyzing expenses and income. Represented in a dollar amount.
8. Funds: The total funds raised for the syndication including Sponsor contribution, investor contribution, and loan contribution.

Relationships:

1. A general manager **manages** syndications: A general manager may be in charge of multiple syndications. Call this relationship **manage_synd.**
2. A general manager **manages** prospective acquisitions: A general manager may be in charge of multiple syndications. Call this relationship **manage_acq.**
3. A general manager **manages** loans: One general manager may manage several loans. Call this relationship manage_loan.
4. A landlord **manages** tenants, lease agreements, and properties: One landlord can manage one property, the property's tenants, and the tenants' lease agreements. This relationship is called **manage_landlord.**
5. An employee **works at** an office: An employee can work at exactly one office. An office has several employees who work there.
6. A Pro Forma **uses** rent roll, expenses per unit, unit mix and rent schedule, expenses seller pro forma, expenses prior year actuals, projected expenses, net operating income: One pro forma uses one of each of the aforementioned tables. Call these relations **uses_roll, uses_mix_schedule, uses_expense_per, uses_prior_actuals, uses_NOI, uses_projected, uses_seller.**
7. A prospective acquisition **has** a pro forma: Each prospective acquisition has exactly one Pro forma.
8. A general manager **is an** employee. Call this relationship **is_general.**
9. A landlord **is an** employee**.** Call this relationship **is_landlord.**
10. A tenant **lives** at a current property: One tenant lives at one current property.
11. A tenant **signs** a lease agreement: one tenant signs one lease agreement.

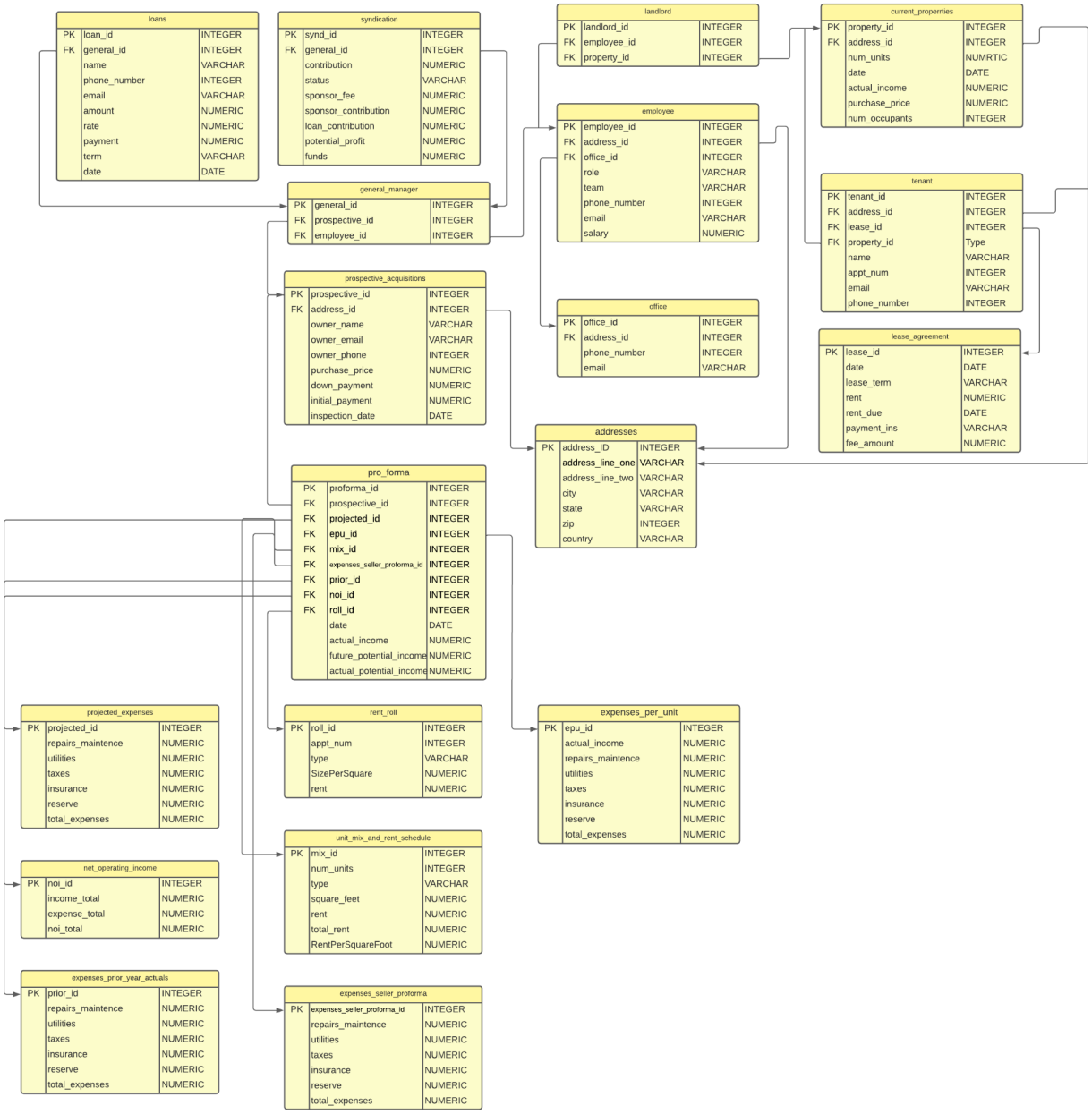
(17 Relationships)

Entity Relationship Schema



Link to view on lucid charts:
https://lucid.app/lucidchart/1e20925f-0fd4-4127-aef8-0ab3cb840b78/edit?invitationId=inv_dd517ae9-e9e5-4bb1-999c-dee8e4aa5e50

Relational Schema


loans
PK loan_id INTEGER
FK general_id INTEGER
name VARCHAR
phone_number INTEGER
email VARCHAR
amount NUMERIC
rate NUMERIC
payment NUMERIC
term VARCHAR
date DATE
syndication
PK synd_id INTEGER
FK general_id INTEGER
contribution NUMERIC
status VARCHAR
sponsor_fee NUMERIC
sponsor_contribution NUMERIC
loan_contribution NUMERIC
potential_profit NUMERIC
funds NUMERIC
landlord
PK landlord_id INTEGER
FK employee_id INTEGER
FK property_id INTEGER
current_properrties
PK property_id INTEGER
FK address_id INTEGER
num_units NUMRTIC
date DATE
actual_income NUMERIC
purchase_price NUMERIC
num_occupants INTEGER
employee
PK employee_id INTEGER
FK address_id INTEGER
FK office_id INTEGER
role VARCHAR
team VARCHAR
phone_number INTEGER
email VARCHAR
salary NUMERIC
general_manager
PK general_id INTEGER
FK prospective_id INTEGER
FK employee_id INTEGER
tenant
PK tenant_id INTEGER
FK address_id INTEGER
FK lease_id INTEGER
FK property_id Type
name VARCHAR
appt_num INTEGER
email VARCHAR
phone_number INTEGER
prospective_acquisitions
PK prospective_id INTEGER
FK address_id INTEGER
owner_name VARCHAR
owner_email VARCHAR
owner_phone INTEGER
purchase_price NUMERIC
down_payment NUMERIC
initial_payment NUMERIC
inspection_date DATE
office
PK office_id INTEGER
FK address_id INTEGER
phone_number INTEGER
email VARCHAR
lease_agreement
PK lease_id INTEGER
date DATE
lease_term VARCHAR
rent NUMERIC
rent_due DATE
payment_ins VARCHAR
fee_amount NUMERIC
addresses
PK address_ID INTEGER
address_line_one VARCHAR
address_line_two VARCHAR
city VARCHAR
state VARCHAR
zip INTEGER
country VARCHAR
pro_forma
PK proforma_id INTEGER
FK prospective_id INTEGER
FK projected_id INTEGER
FK epu_id INTEGER
FK mix_id INTEGER
FK expenses_seller_proforma_id INTEGER
FK prior_id INTEGER
FK noi_id INTEGER
FK roll_id INTEGER
date DATE
actual_income NUMERIC
future_potential_income NUMERIC
actual_potential_income NUMERIC
projected_expenses
PK projected_id INTEGER
repairs_maintence NUMERIC
utilities NUMERIC
taxes NUMERIC
insurance NUMERIC
reserve NUMERIC
total_expenses NUMERIC
rent_roll
PK roll_id INTEGER
appt_num INTEGER
type VARCHAR
SizePerSquare NUMERIC
rent NUMERIC
expenses_per_unit
PK epu_id INTEGER
actual_income NUMERIC
repairs_maintence NUMERIC
utilities NUMERIC
taxes NUMERIC
insurance NUMERIC
reserve NUMERIC
total_expenses NUMERIC
unit_mix_and_rent_schedule
PK mix_id INTEGER
num_units INTEGER
type VARCHAR
square_feet NUMERIC
rent NUMERIC
total_rent NUMERIC
RentPerSquareFoot NUMERIC
net_operating_income
PK noi_id INTEGER
income_total NUMERIC
expense_total NUMERIC
noi_total NUMERIC
expenses_prior_year_actuals
PK prior_id INTEGER
repairs_maintence NUMERIC
utilities NUMERIC
taxes NUMERIC
insurance NUMERIC
reserve NUMERIC
total_expenses NUMERIC
expenses_seller_proforma
PK expenses_seller_proforma_id INTEGER
repairs_maintence NUMERIC
utilities NUMERIC
taxes NUMERIC
insurance NUMERIC
reserve NUMERIC
total_expenses NUMERIC

DDL

```
CREATE TABLE "expenses_per_unit" (
  "epu_id" INTEGER,
  "actual_income" NUMERIC(11,2) DEFAULT 0.00,
  "repairs_maintence" NUMERIC(11,2) DEFAULT 0.00,
  "utilities" NUMERIC(11,2) DEFAULT 0.00,
  "taxes" NUMERIC(11,2) DEFAULT 0.00,
  "insurance" NUMERIC(11,2) DEFAULT 0.00,
  "reserve" NUMERIC(11,2) DEFAULT 0.00,
  "total_expenses" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "expenses_per_unit_pkey" PRIMARY KEY ("epu_id")
);


CREATE TABLE "net_operating_income" (
  "noi_id" INTEGER,
  "income_total" NUMERIC(11,2) DEFAULT 0.00,
  "expense_total" NUMERIC(11,2) DEFAULT 0.00,
  "noi_total" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "net_operating_income_pkey" PRIMARY KEY ("noi_id")
);

CREATE TABLE "projected_expenses" (
  "projected_id" INTEGER,
  "repairs_maintence" NUMERIC(11,2) DEFAULT 0.00,
  "utilities" NUMERIC(11,2) DEFAULT 0.00,
  "taxes" NUMERIC(11,2) DEFAULT 0.00,
  "insurance" NUMERIC(11,2) DEFAULT 0.00,
  "reserve" NUMERIC(11,2) DEFAULT 0.00,
  "total_expenses" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "projected_expenses_pkey" PRIMARY KEY ("projected_id")
);

CREATE TABLE "current_properties" (
  "property_id" INTEGER,
  "address_id" INTEGER,
  "num_units" NUMERIC(11,2) DEFAULT 0.00,
  "date" DATE,
  "actual_income" NUMERIC(11,2) DEFAULT 0.00,
  "purchase_price" NUMERIC(11,2) DEFAULT 0.00,
  "num_occupants" INTEGER,
  CONSTRAINT "current_properties_pkey" PRIMARY KEY ("property_id")
);
```

```
CREATE TABLE "syndication" (
  "synd_id" INTEGER,
  "general_id" INTEGER,
  "contribution" NUMERIC(11,2) DEFAULT 0.00,
  "status" VARCHAR(50),
  "sponsor_fee" NUMERIC(11,2) DEFAULT 0.00,
  "sponsor_contribution" NUMERIC(11,2) DEFAULT 0.00,
  "loan_contribution" NUMERIC(11,2) DEFAULT 0.00,
  "potential_profit" NUMERIC(11,2) DEFAULT 0.00,
  "funds" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "syndication_pkey" PRIMARY KEY ("synd_id")
);

CREATE TABLE "unit_mix_and_rent_schedule" (
  "mix_id" INTEGER,
  "num_units" INTEGER,
  "type" VARCHAR(50),
  "square_feet" NUMERIC(11,2) DEFAULT 0.00,
  "rent" NUMERIC(11,2) DEFAULT 0.00,
  "total_rent" NUMERIC(11,2) DEFAULT 0.00,
  "rent_per_square_foot" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "unit_mix_and_rent_schedule_pkey" PRIMARY KEY ("mix_id")

);

CREATE TABLE "expenses_prior_year_actuals" (
  "prior_id" INTEGER,
  "repairs_maintence" NUMERIC(11,2) DEFAULT 0.00,
  "utilities" NUMERIC(11,2) DEFAULT 0.00,
  "taxes" NUMERIC(11,2) DEFAULT 0.00,
  "insurance" NUMERIC(11,2) DEFAULT 0.00,
  "reserve" NUMERIC(11,2) DEFAULT 0.00,
  "total_expenses" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "expenses_prior_year_actuals_pkey" PRIMARY KEY ("prior_id")
);

CREATE TABLE "rent_roll" (
  "roll_id" INTEGER,
  "appt_num" INTEGER,
  "type" VARCHAR(50),
  "size_per_square" NUMERIC(11,2) DEFAULT 0.00,
  "rent" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "rent_roll_pkey" PRIMARY KEY ("roll_id")
);
```

```
CREATE TABLE "expenses_seller_proforma" (
  "expenses_seller_proforma_id" INTEGER,
  "repairs_maintence" NUMERIC(11,2) DEFAULT 0.00,
  "utilities" NUMERIC(11,2) DEFAULT 0.00,
  "taxes" NUMERIC(11,2) DEFAULT 0.00,
  "insurance" NUMERIC(11,2) DEFAULT 0.00,
  "reserve" NUMERIC(11,2) DEFAULT 0.00,
  "total_expenses" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "expenses_seller_proforma_pkey" PRIMARY KEY
("expenses_seller_proforma_id")
);

CREATE TABLE "addresses" (
  "address_id" INTEGER,
  "address_line_one" VARCHAR(50),
  "address_line_two" VARCHAR(50),
  "city" VARCHAR(50),
  "state" VARCHAR(50),
  "zip" INTEGER,
  "country" VARCHAR(50),
  CONSTRAINT "addresses_pkey" PRIMARY KEY ("address_id")
);

CREATE TABLE "office" (
  "office_id" INTEGER,
  "address_id" INTEGER,
  "phone_number" INTEGER,
  "email" VARCHAR(50),
  CONSTRAINT "office_pkey" PRIMARY KEY ("office_id"),
  CONSTRAINT "office_fkey" FOREIGN KEY ("address_id") REFERENCES "addresses"
("address_id")
    ON DELETE SET NULL
);

CREATE TABLE "prospective_acquisitions" (
  "prospective_id" INTEGER,
  "address_id" INTEGER,
  "owner_name" VARCHAR(50) NOT NULL,
  "owner_email" VARCHAR(50) NOT NULL,
  "owner_phone" INTEGER NOT NULL,
  "purchase_price" NUMERIC(11,2) DEFAULT 0.00,
  "down_payment" NUMERIC(11,2) DEFAULT 0.00,
  "initial_payment" NUMERIC(11,2) DEFAULT 0.00,
```

```
  "inspection_date" DATE,
  CONSTRAINT "prospective_acquisitions_pkey" PRIMARY KEY ("prospective_id"),
  CONSTRAINT "prospective_acquisitions_fkey" FOREIGN KEY ("address_id") REFERENCES
"addresses" ("address_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE "lease_agreement" (
  "lease_id" INTEGER,
  "date" DATE,
  "lease_term" VARCHAR(50),
  "rent" NUMERIC(11,2) DEFAULT 0.00,
  "rent_due" DATE,
  "payment_ins" VARCHAR(50),
  "fee_amount" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "lease_agreement_pkey" PRIMARY KEY ("lease_id")
);

CREATE TABLE "employee" (
  "employee_id" INTEGER,
  "address_id" INTEGER,
  "office_id" INTEGER,
  "role" VARCHAR(50),
  "team" VARCHAR(50),
  "phone_number" INTEGER NOT NULL,
  "email" VARCHAR(50) NOT NULL,
  "salary" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "employee_pkey" PRIMARY KEY ("employee_id"),
  CONSTRAINT "employee_fkey1" FOREIGN KEY ("office_id") REFERENCES office
("office_id")
        ON UPDATE CASCADE
        ON DELETE CASCADE,
  CONSTRAINT "employee_fkey2" FOREIGN KEY ("address_id") REFERENCES "addresses"
("address_id")
        ON UPDATE CASCADE
        ON DELETE CASCADE
);

CREATE TABLE "general_manager" (
  "general_id" INTEGER,
  "prospective_id" INTEGER,
  "employee_id" INTEGER,
  CONSTRAINT "general_manager_pkey" PRIMARY KEY ("general_id"),
```

```
  CONSTRAINT "general_manager_fkey1" FOREIGN KEY ("prospective_id") REFERENCES
"prospective_acquisitions" ("prospective_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT "general_manager_fkey2" FOREIGN KEY ("employee_id") REFERENCES
"employee" ("employee_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE "tenant" (
  "tenant_id" INTEGER,
  "address_id" INTEGER,
  "lease_id" INTEGER,
  "property_id" INTEGER,
  "name" VARCHAR(50) NOT NULL,
  "appt_num" INTEGER,
  "email" VARCHAR(50) NOT NULL,
  "phone_number" INTEGER NOT NULL,
  CONSTRAINT "tenant_pkey" PRIMARY KEY ("tenant_id"),
  CONSTRAINT "tenant_fkey" FOREIGN KEY ("address_id") REFERENCES "addresses"
("address_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT "tenant_fkey2" FOREIGN KEY ("lease_id") REFERENCES "lease_agreement"
("lease_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT "tenant_fkey3" FOREIGN KEY ("property_id") REFERENCES
"current_properties" ("property_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE "landlord" (
  "landlord_id" INTEGER,
  "employee_id" INTEGER,
  "property_id" INTEGER,
  CONSTRAINT "landlord_pkey" PRIMARY KEY ("landlord_id"),
  CONSTRAINT "landlord_fkey1" FOREIGN KEY ("employee_id") REFERENCES "employee"
("employee_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
```

```
  CONSTRAINT "landlord_fkey2" FOREIGN KEY ("property_id") REFERENCES
"current_properties" ("property_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE "loans" (
  "loan_id" INTEGER,
  "general_id" INTEGER,
  "name" VARCHAR(50) NOT NULL,
  "phone_number" INTEGER,
  "email" VARCHAR(50),
  "amount" NUMERIC(11,2) DEFAULT 0.00,
  "rate" NUMERIC(11,2) DEFAULT 0.00,
  "payment" NUMERIC(11,2) DEFAULT 0.00,
  "term" VARCHAR(50),
  "date" DATE,
  CONSTRAINT "loans_pkey" PRIMARY KEY ("loan_id"),
  CONSTRAINT "loans_fkey" FOREIGN KEY ("general_id") REFERENCES "general_manager"
("general_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE
);

CREATE TABLE "pro_forma" (
  "proforma_id" INTEGER,
  "prospective_id" INTEGER,
  "projected_id" INTEGER,
  "expenses_seller_proforma_id" integer,
  "epu_id" INTEGER,
  "mix_id" INTEGER,
  "prior_id" INTEGER,
  "noi_id" INTEGER,
  "roll_id" INTEGER,
  "date" DATE,
  "actual_income" NUMERIC(11,2) DEFAULT 0.00,
  "future_potential_income" NUMERIC(11,2) DEFAULT 0.00,
  "actual_potential_income" NUMERIC(11,2) DEFAULT 0.00,
  CONSTRAINT "pro_forma_pkey"  PRIMARY KEY ("proforma_id"),
  CONSTRAINT  "pro_forma_fkey" FOREIGN KEY ("prospective_id") REFERENCES
"prospective_acquisitions" ("prospective_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
```

```
  CONSTRAINT "pro_forma_fkey2" FOREIGN KEY ("projected_id") REFERENCES
"projected_expenses" ("projected_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT "pro_forma_fkey3" FOREIGN KEY ("epu_id") REFERENCES
"expenses_per_unit" ("epu_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT "pro_forma_fkey4" FOREIGN KEY ("mix_id") REFERENCES
"unit_mix_and_rent_schedule" ("mix_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT "pro_forma_fkey5" FOREIGN KEY ("expenses_seller_proforma_id")
REFERENCES "expenses_seller_proforma" ("expenses_seller_proforma_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT "pro_forma_fkey6" FOREIGN KEY ("prior_id") REFERENCES
"expenses_prior_year_actuals" ("prior_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT "pro_forma_fkey7" FOREIGN KEY ("noi_id") REFERENCES
"net_operating_income" ("noi_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT "pro_forma_fkey8" FOREIGN KEY ("roll_id") REFERENCES "rent_roll"
("roll_id")
    ON DELETE CASCADE
    ON UPDATE CASCADE
);
```

Insert Statements

```
--current_properties
INSERT INTO current_properties ("property_id", "address_id", "num_units", "date",
"actual_income", "purchase_price", "num_occupants")
VALUES (85610 , 73332 , 408 , '2000-12-31', 150000 , 2052033 , 638 );
INSERT INTO current_properties ("property_id", "address_id", "num_units", "date",
"actual_income", "purchase_price", "num_occupants")
VALUES (68262 ,73734 ,402 , '2018-06-06'  ,9361602 ,6093446 ,453 );
INSERT INTO current_properties ("property_id", "address_id", "num_units", "date",
"actual_income", "purchase_price", "num_occupants")
 VALUES (35574 ,91462 ,331 , '2020-06-04' ,6269812 ,4491732 ,925 );
INSERT INTO current_properties ("property_id", "address_id", "num_units", "date",
"actual_income", "purchase_price", "num_occupants")
```

```
VALUES (97587 ,85844 ,160 , '2020-06-04'  ,5149029 ,9631141 ,32 );
INSERT INTO current_properties ("property_id", "address_id", "num_units", "date",
"actual_income", "purchase_price", "num_occupants")
VALUES (98385 ,56249 ,22 , '2020-06-04',4803640 ,8241835 ,478 );
INSERT INTO current_properties ("property_id", "address_id", "num_units", "date",
"actual_income", "purchase_price", "num_occupants")
VALUES (86534 ,13563 ,133 , '2020-06-04' ,570802 ,1553028 ,315 );
INSERT INTO current_properties ("property_id", "address_id", "num_units", "date",
"actual_income", "purchase_price", "num_occupants")
VALUES (86268 ,89913 ,347 , '2020-06-04' ,442143, 100000 ,154 );
INSERT INTO current_properties ("property_id", "address_id", "num_units", "date",
"actual_income", "purchase_price", "num_occupants")
VALUES (93292 ,13069 ,84 , '2020-06-04' ,8768809 ,2281646 ,907 );

--expenses per unit
INSERT INTO expenses_per_unit ("epu_id", "actual_income", "repairs_maintence", "utilities",
"taxes", "insurance", "reserve", "total_expenses")
VALUES (72945 ,45282242 ,2282121 ,4397490 ,345578920 ,6355301 ,318609 ,399009 );
INSERT INTO expenses_per_unit ("epu_id", "actual_income", "repairs_maintence", "utilities",
"taxes", "insurance", "reserve", "total_expenses")
VALUES (50394 ,85094482 ,3897567 ,5681965 ,9987307 ,577752 ,457056 ,456036 );
INSERT INTO expenses_per_unit ("epu_id", "actual_income", "repairs_maintence", "utilities",
"taxes", "insurance", "reserve", "total_expenses")
VALUES (25475 ,48318018 ,3944500 ,4273044 ,6969067 ,5227439 ,6431559 ,446834 );
INSERT INTO expenses_per_unit ("epu_id", "actual_income", "repairs_maintence", "utilities",
"taxes", "insurance", "reserve", "total_expenses")
VALUES (28775 ,7967977 ,8228039 ,1643539 ,414359 ,4809462 ,2757049 ,527375 );
INSERT INTO expenses_per_unit ("epu_id", "actual_income", "repairs_maintence", "utilities",
"taxes", "insurance", "reserve", "total_expenses")
VALUES (72129 ,573743 ,9845497 ,6116053 ,3753996 ,9270883 ,9003021 ,9267917 );
INSERT INTO expenses_per_unit ("epu_id", "actual_income", "repairs_maintence", "utilities",
"taxes", "insurance", "reserve", "total_expenses")
VALUES (58703 ,7986318 ,7244857 ,4548293 ,3837109 ,1389016 ,6172456 ,3886421 );
INSERT INTO expenses_per_unit ("epu_id", "actual_income", "repairs_maintence", "utilities",
"taxes", "insurance", "reserve", "total_expenses")
VALUES (99658 ,939833 ,291660 ,3731874 ,4999955 ,3177127 ,2486649 ,389653 );

---rent roll_id
INSERT INTO rent_roll ("roll_id", "appt_num", "type", "size_per_square", "rent")
VALUES (80868 ,142 ,'2/1' ,1851 ,1035 );
INSERT INTO rent_roll ("roll_id", "appt_num", "type", "size_per_square", "rent")
VALUES (28352 ,272 ,'3/2' ,1945 ,1034 );
INSERT INTO rent_roll ("roll_id", "appt_num", "type", "size_per_square", "rent")
VALUES (95873 ,449 ,'3/2' ,1613 ,498 );
```

```
INSERT INTO rent_roll ("roll_id", "appt_num", "type", "size_per_square", "rent")
VALUES (31350 ,305 ,'3/2' ,711 ,878 );
INSERT INTO rent_roll ("roll_id", "appt_num", "type", "size_per_square", "rent")
VALUES (32136 ,107 ,'2/1' ,650 ,1091 );
INSERT INTO rent_roll ("roll_id", "appt_num", "type", "size_per_square", "rent")
VALUES (52920 ,443 ,'1/1' ,357 ,468 );

--unit_mix_and_rent_schedule
INSERT INTO unit_mix_and_rent_schedule ("mix_id", "num_units", "type", "square_feet", "rent",
"total_rent", "rent_per_square_foot")
VALUES (55404 ,199 , '2/1' ,2500 ,1035 ,205965 ,0.414 );
INSERT INTO unit_mix_and_rent_schedule ("mix_id", "num_units", "type", "square_feet", "rent",
"total_rent", "rent_per_square_foot")
VALUES (63084 ,490 , '1/1' ,700 ,468 ,229320 ,0.668571428571429 );
INSERT INTO unit_mix_and_rent_schedule ("mix_id", "num_units", "type", "square_feet", "rent",
"total_rent", "rent_per_square_foot")
VALUES (79357 ,557 , '3/2' ,1736 ,1034 ,575938 ,0.595622119815668 );
INSERT INTO unit_mix_and_rent_schedule ("mix_id", "num_units", "type", "square_feet", "rent",
"total_rent", "rent_per_square_foot")
VALUES (88556 ,271 ,'3/2' ,1534 ,498 ,134958 ,0.324641460234681 );
INSERT INTO unit_mix_and_rent_schedule ("mix_id", "num_units", "type", "square_feet", "rent",
"total_rent", "rent_per_square_foot")
VALUES (52557 ,294 , '2/1' ,903 ,1091 ,320754 ,1.20819490586932 );
INSERT INTO unit_mix_and_rent_schedule ("mix_id", "num_units", "type", "square_feet", "rent",
"total_rent", "rent_per_square_foot")
VALUES (24966 ,295 , '3/2' ,1633 ,878 ,259010 ,0.537660747091243 );

---expenses_seller_proforma
INSERT INTO expenses_seller_proforma ("expenses_seller_proforma_id","repairs_maintence",
"utilities", "taxes", "insurance", "reserve", "total_expenses")
VALUES (44522 ,86956005 ,57451945 ,52475995 ,95154714 ,6941105 ,76471437);
INSERT INTO expenses_seller_proforma ("expenses_seller_proforma_id","repairs_maintence",
"utilities", "taxes", "insurance", "reserve", "total_expenses")
VALUES (25010 ,6938986 ,64789397 ,63959230 ,88365519 ,82643254 ,57398674);
INSERT INTO expenses_seller_proforma ("expenses_seller_proforma_id","repairs_maintence",
"utilities", "taxes", "insurance", "reserve", "total_expenses")
VALUES (63882 ,64510902 ,39781363 ,25169834 ,29752390 ,49918255 ,5543830);
INSERT INTO expenses_seller_proforma ("expenses_seller_proforma_id","repairs_maintence",
"utilities", "taxes", "insurance", "reserve", "total_expenses")
VALUES (29215 ,34805812 ,6619389 ,53663563 ,11232504 ,7027338 ,99749795 );
INSERT INTO expenses_seller_proforma ("expenses_seller_proforma_id","repairs_maintence",
"utilities", "taxes", "insurance", "reserve", "total_expenses")
VALUES (16078 ,74830072 ,25006557 ,12388385 ,91375077 ,12698023 ,13702674);
```

```
INSERT INTO expenses_seller_proforma ("expenses_seller_proforma_id","repairs_maintence",
"utilities", "taxes", "insurance", "reserve", "total_expenses")
VALUES (36479 ,77731303 ,61770888 ,99198709 ,14818199 ,74377898 ,27220958);

---expenses_prior_year_actuals
INSERT INTO expenses_prior_year_actuals ("prior_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (67686 ,6938986 ,64789397 ,63959230 ,88365519 ,82643254 ,57398674 );
INSERT INTO expenses_prior_year_actuals ("prior_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (12019 ,64510902 ,3971363 ,25169834 ,29752390 ,49918255 ,5543830  );
INSERT INTO expenses_prior_year_actuals ("prior_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (40787 ,34805812 ,6619389 ,53663563 ,11232504 ,7027338 ,99749795  );
INSERT INTO expenses_prior_year_actuals ("prior_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (89921 ,74830072 ,25006557 ,12388385 ,913750 ,12698023 ,13702674 );


---net_operating_income
INSERT INTO net_operating_income ("noi_id", "income_total", "expense_total", "noi_total")
VALUES (41091 ,86288090 ,76471437 ,9816653 );
VALUES (29504 ,16681941 ,57398674 ,-40716733 );
INSERT INTO net_operating_income ("noi_id", "income_total", "expense_total", "noi_total")
VALUES (46306 ,15034508 ,5543830 ,9490678 );
INSERT INTO net_operating_income ("noi_id", "income_total", "expense_total", "noi_total")
VALUES (27938 ,58227046 ,99749795 ,-41522749 );
INSERT INTO net_operating_income ("noi_id", "income_total", "expense_total", "noi_total")
VALUES (56197 ,212970950 ,13702674 ,199268276 );
INSERT INTO net_operating_income ("noi_id", "income_total", "expense_total", "noi_total")
VALUES (23027 ,138401770 ,27220958 ,111180812 );


---projected_expenses
INSERT INTO projected_expenses ("projected_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (62895 ,815432 ,5827355 ,7792469 ,9274737 ,5900504 ,9003021 );
INSERT INTO projected_expenses ("projected_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (39473 ,6344320 ,9995579 ,9264586 ,3917308 ,5260462 ,9080914 );
INSERT INTO projected_expenses ("projected_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (80280 ,9471858 ,1240018 ,8672753 ,3907153 ,7569609 ,6172456 );
```

```
INSERT INTO projected_expenses ("projected_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (85795 ,306181 ,2178480 ,9337593 ,9678576 ,9459941 ,178373039 );
INSERT INTO projected_expenses ("projected_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (75907  ,2740986 ,788698 ,1992959 ,1022443 ,5971582 ,943365909 );
INSERT INTO projected_expenses ("projected_id", "repairs_maintence", "utilities", "taxes",
"insurance", "reserve", "total_expenses")
VALUES (59847 ,1866051 ,197475 ,1298859 ,579975 ,4867403 , 591684172 );

--addresses
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (49445 ,'14755 Ventura Boulevard' ,'1-70' ,'Sherman Oaks' ,'CA' ,91403 ,'United States
of America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (94373 ,'641 Lexington Avenue' ,'Suite 1400' ,'Ney York' ,'NY' ,10022 ,'United States of
America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (73816 ,'1122 Robertson Boulevard' ,'#15' ,'Los Angeles' ,'CA' ,90035 ,'United States
of America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (73205 ,'930 5th Ave' ,'0' ,'New York' ,'NY' ,90035 ,'United States of America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (20870 ,'132 Rodeo Drive' ,'#300' ,'Beverly Hills' ,'CA' ,90212 ,'United States of
America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (29420 ,'8033 Sunset Boulevard' ,'#605' ,'Los Angeles' ,'CA' ,90046 ,'United States of
America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (57423 ,'1828 Courtney Avenue' ,'0' ,'Los Angeles' ,'CA' ,90046 ,'United States of
America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (58110 ,'955 S. Carrillo Drive' ,'#200' ,'Los Angeles' ,'CA' ,90048 ,'United States of
America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
```

VALUES (79320 ,'123 Rodeo Drive' ,'#300' ,'Beverly Hills' ,'CA' ,90212 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (92816 ,'433 Forman Ave' ,'0' ,'Toluca Lake' ,'CA' ,91602 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (21715 ,'1180 S. Beverly Drive' ,'#618' ,'Los Angeles' ,'CA' ,90038 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (22827 ,'555 Melrose Avenue' ,'#371' ,'Los Angeles' ,'CA' ,90038 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (84944 ,'12400 Ventura Boulevard' ,'#400' ,'Studio City' ,'CA' ,91604 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (81036 ,'14755 Ventura Boulevard' ,'1-70' ,'Sherman Oaks' ,'CA' ,91403 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (30473 ,'641 Lexington Avenue' ,'Suite 1400' ,'Ney York' ,'NY' ,10022 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (92136 ,'1122 Robertson Boulevard' ,'#15' ,'Los Angeles' ,'CA' ,90035 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (97301 ,'930 5th Ave' ,'0' ,'New York' ,'NY' ,90035 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (57189 ,'132 Rodeo Drive' ,'#300' ,'Beverly Hills' ,'CA' ,90212 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (60033 ,'8033 Sunset Boulevard' ,'#605' ,'Los Angeles' ,'CA' ,90046 ,'United States of America');
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state", "zip", "country")
VALUES (44069 ,'955 S. Carrillo Drive' ,'#200' ,'Los Angeles' ,'CA' ,90048 ,'United States of America');

```
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (82520 ,'123 Rodeo Drive' ,'#300' ,'Beverly Hills' ,'CA' ,90212 ,'United States of
America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (33062 ,'433 Forman Ave' ,'0' ,'Toluca Lake' ,'CA' ,91602 ,'United States of America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (70498 ,'1180 S. Beverly Drive' ,'#618' ,'Los Angeles' ,'CA' ,90038 ,'United States of
America' );
INSERT INTO addresses("address_id", "address_line_one", "address_line_two", "city", "state",
"zip", "country")
VALUES (35951 ,'555 Melrose Avenue' ,'#371' ,'Los Angeles' ,'CA' ,90038 ,'United States of
America' );

--office
INSERT INTO office ("office_id", "address_id", "phone_number", "email")
VALUES (28227 ,49445 ,423338149 ,'faker365@hotmail.com' );
INSERT INTO office ("office_id", "address_id", "phone_number", "email")
VALUES (86286 ,94373 ,867750631 ,'faker365@hotmail.com' );
INSERT INTO office ("office_id", "address_id", "phone_number", "email")
VALUES (83973 ,73816 ,947220875 ,'faker365@hotmail.com' );
INSERT INTO office ("office_id", "address_id", "phone_number", "email")
VALUES (21684 ,73205 ,550656406 ,'faker365@hotmail.com' );
INSERT INTO office ("office_id", "address_id", "phone_number", "email")
VALUES (62930 ,20870 ,296619627 ,'faker365@hotmail.com' );
INSERT INTO office ("office_id", "address_id", "phone_number", "email")
VALUES (16149 ,29420 ,798300968 ,'faker365@hotmail.com' );

--lease_agreement
INSERT INTO lease_agreement ("lease_id", "date", "lease_term", "rent", "rent_due",
"payment_ins", "fee_amount")
VALUES (70380 , '2015-08-28' ,'year-to-year' ,100200 , '2015-08-28' ,'0' ,50 );
INSERT INTO lease_agreement ("lease_id", "date", "lease_term", "rent", "rent_due",
"payment_ins", "fee_amount")
VALUES (50810 , '2014-05-13' ,'month-to-month' ,154200 , '2014-05-13' ,'0' ,60 );
INSERT INTO lease_agreement ("lease_id", "date", "lease_term", "rent", "rent_due",
"payment_ins", "fee_amount")
VALUES (63053 ,'2018-07-05' ,'year-to-year' ,90600 , '2018-07-05' ,'0' ,70 );
INSERT INTO lease_agreement ("lease_id", "date", "lease_term", "rent", "rent_due",
"payment_ins", "fee_amount")
VALUES (65786 , '2015-08-28' ,'year-to-year' ,97000 , '2015-08-28' ,'0' ,80 );
```

```
INSERT INTO lease_agreement ("lease_id", "date", "lease_term", "rent", "rent_due",
"payment_ins", "fee_amount")
VALUES (93624 ,'2014-05-13' ,'month-to-month' ,83300 , '2014-05-13' ,'0' ,80 );
INSERT INTO lease_agreement ("lease_id", "date", "lease_term", "rent", "rent_due",
"payment_ins", "fee_amount")
VALUES (93408 , '2018-07-05' ,'year-to-year' ,84400 , '2018-07-05' ,'0' ,90 );

--prospective_acquisitions
INSERT INTO prospective_acquisitions ("prospective_id", "address_id", "owner_name",
"owner_email", "owner_phone", "purchase_price", "down_payment", "initial_payment",
"inspection_date")
VALUES (43906 ,57423 ,'Lady Gaga' ,'faker6754@gmail.com' ,55373354 ,166898739 ,797361
,79361 ,'2022-06-07' );
INSERT INTO prospective_acquisitions ("prospective_id", "address_id", "owner_name",
"owner_email", "owner_phone", "purchase_price", "down_payment", "initial_payment",
"inspection_date")
VALUES (64317 ,58110 ,'Katy Perry' ,'faker6754@gmail.com' ,11268476 ,767297308 ,94853
,9485 ,'2022-06-07' );
INSERT INTO prospective_acquisitions ("prospective_id", "address_id", "owner_name",
"owner_email", "owner_phone", "purchase_price", "down_payment", "initial_payment",
"inspection_date")
VALUES (20030 ,79320 ,'Justin Timberlake' ,'faker6754@gmail.com' ,5392758 ,65694653
,312781 ,31278 ,'2022-06-07' );
INSERT INTO prospective_acquisitions ("prospective_id", "address_id", "owner_name",
"owner_email", "owner_phone", "purchase_price", "down_payment", "initial_payment",
"inspection_date")
VALUES (60711 ,92816 ,'David Letterman' ,'faker6754@gmail.com' ,52465676 ,189823011
,267603 ,27603 ,'2022-06-07' );
INSERT INTO prospective_acquisitions ("prospective_id", "address_id", "owner_name",
"owner_email", "owner_phone", "purchase_price", "down_payment", "initial_payment",
"inspection_date")
VALUES (32679 ,21715 ,'Jennifer Aniston' ,'faker6754@gmail.com' ,57022628 ,54651756
,690804 ,60804 ,'2022-06-07' );
INSERT INTO prospective_acquisitions ("prospective_id", "address_id", "owner_name",
"owner_email", "owner_phone", "purchase_price", "down_payment", "initial_payment",
"inspection_date")
VALUES (32591 ,22827 ,'Donald Duck' ,'faker6754@gmail.com' ,34869501 ,881252000
,287892 ,7892 ,'2022-06-07' );

--employee
INSERT INTO employee ("employee_id", "address_id", "office_id", "role", "team",
"phone_number", "email", "salary")
VALUES (55770 ,84944 ,28227 ,'Accountant' ,'Accounting and Finance' ,55373535
,'faker6754@gmail.com' ,18332908 );
```

```
INSERT INTO employee ("employee_id", "address_id", "office_id", "role", "team",
"phone_number", "email", "salary")
VALUES (67495 ,81036 ,86286 ,'Attorney' ,'Law and Taxes' ,11266847 ,'faker6754@gmail.com'
,72815159 );
INSERT INTO employee ("employee_id", "address_id", "office_id", "role", "team",
"phone_number", "email", "salary")
VALUES (85225 ,30473 ,83973 ,'Attorney' ,'Law and Taxes' ,58397275 ,'faker6754@gmail.com'
,16732992 );
INSERT INTO employee ("employee_id", "address_id", "office_id", "role", "team",
"phone_number", "email", "salary")
VALUES (63779 ,92136 ,21684 ,'Maintence Tech' ,'Management Team' ,52646567
,'faker6754@gmail.com' ,98097728 );
INSERT INTO employee ("employee_id", "address_id", "office_id", "role", "team",
"phone_number", "email", "salary")
VALUES (71128 ,97301 ,62930 ,'Broker' ,'Real Estate' ,50702262 ,'faker6754@gmail.com'
,35228981 );
INSERT INTO employee ("employee_id", "address_id", "office_id", "role", "team",
"phone_number", "email", "salary")
VALUES (85622 ,57189 ,16149 ,'Intern' ,'Administration' ,34869750 ,'faker6754@gmail.com'
,35174122 );

--general_manager
INSERT INTO general_manager ("general_id", "prospective_id", "employee_id")
VALUES (58659 ,43906 ,55770 );
INSERT INTO general_manager ("general_id", "prospective_id", "employee_id")
VALUES (52895 ,64317 ,67495 );
INSERT INTO general_manager ("general_id", "prospective_id", "employee_id")
VALUES (96251 ,20030 ,85225 );
INSERT INTO general_manager ("general_id", "prospective_id", "employee_id")
VALUES (23353 ,60711 ,63779 );
INSERT INTO general_manager ("general_id", "prospective_id", "employee_id")
VALUES (82711 ,32679 ,71128 );
INSERT INTO general_manager ("general_id", "prospective_id", "employee_id")
VALUES (45782 ,32591 ,85622 );

-- syndication
INSERT INTO syndication ("synd_id", "general_id", "contribution", "status", "sponsor_fee",
"sponsor_contribution", "loan_contribution", "potential_profit", "funds")
VALUES (13433 ,58659 ,894180 ,'Enter Status' ,178836 ,415787 ,66868 ,800222516 ,1376835
);
INSERT INTO syndication ("synd_id", "general_id", "contribution", "status", "sponsor_fee",
"sponsor_contribution", "loan_contribution", "potential_profit", "funds")
VALUES (73674 ,52895 ,692771 ,'Enter Status' ,138554.2 ,440638 ,8067 ,262401052 ,1141476
);
```

```
INSERT INTO syndication ("synd_id", "general_id", "contribution", "status", "sponsor_fee",
"sponsor_contribution", "loan_contribution", "potential_profit", "funds")
VALUES (15672 ,96251 ,56771 ,'Enter Status' ,11354.2 ,67217 ,91690 ,913288513 ,215678 );
INSERT INTO syndication ("synd_id", "general_id", "contribution", "status", "sponsor_fee",
"sponsor_contribution", "loan_contribution", "potential_profit", "funds")
VALUES (78428 ,23353 ,817294 ,'Enter Status' ,163458.8 ,13465 ,60546 ,228600939 ,891305
);
INSERT INTO syndication ("synd_id", "general_id", "contribution", "status", "sponsor_fee",
"sponsor_contribution", "loan_contribution", "potential_profit", "funds")
VALUES (75515 ,82711 ,448870 ,'Enter Status' ,89774 ,86300 ,18890 ,587550190 ,554060 );
INSERT INTO syndication ("synd_id", "general_id", "contribution", "status", "sponsor_fee",
"sponsor_contribution", "loan_contribution", "potential_profit", "funds")
VALUES (29868 ,45782 ,477693 ,'Enter Status' ,95538.6 ,973236 ,92961 ,842080520 ,1543890
);

--landlord
INSERT INTO landlord ("landlord_id", "employee_id", "property_id")
VALUES (25753 ,55770 ,85610 );
INSERT INTO landlord ("landlord_id", "employee_id", "property_id")
VALUES (62365 ,67495 ,68262 );
INSERT INTO landlord ("landlord_id", "employee_id", "property_id")
VALUES (76140 ,85225 ,35574 );
INSERT INTO landlord ("landlord_id", "employee_id", "property_id")
VALUES (36510 ,63779 ,97587 );
INSERT INTO landlord ("landlord_id", "employee_id", "property_id")
VALUES (27047 ,71128 ,98385 );
INSERT INTO landlord ("landlord_id", "employee_id", "property_id")
VALUES (24387 ,85622 ,86534 );

---tenant
INSERT INTO tenant ("tenant_id", "address_id", "lease_id", "property_id", "name", "appt_num",
"email", "phone_number")
VALUES (53071 ,60033 ,70380 ,85610 ,'Miley Cyrus' ,127 ,'fake124@yahoo.com' ,198152064 );
INSERT INTO tenant ("tenant_id", "address_id", "lease_id", "property_id", "name", "appt_num",
"email", "phone_number")
VALUES (12157 ,44069 ,50810 ,68262 ,'Kim Kardashian' ,342 ,'fake124@yahoo.com'
,233781932 );
INSERT INTO tenant ("tenant_id", "address_id", "lease_id", "property_id", "name", "appt_num",
"email", "phone_number")
VALUES (57234 ,82520 ,63053 ,35574 ,'Kayne West' ,471 ,'fake124@yahoo.com' ,116869022
);
INSERT INTO tenant ("tenant_id", "address_id", "lease_id", "property_id", "name", "appt_num",
"email", "phone_number")
```

```
VALUES (37217 ,33062 ,65786 ,97587 ,'Marget Thatcher' ,38 ,'fake124@yahoo.com'
,688393086 );
INSERT INTO tenant ("tenant_id", "address_id", "lease_id", "property_id", "name", "appt_num",
"email", "phone_number")
VALUES (27031 ,70498 ,93624 ,98385 ,'George Washington' ,459 ,'fake124@yahoo.com'
,396831679 );
INSERT INTO tenant ("tenant_id", "address_id", "lease_id", "property_id", "name", "appt_num",
"email", "phone_number")
VALUES (75874 ,35951 ,93408 ,86534 ,'Ghandi' ,349 ,'fake124@yahoo.com' ,928307244 );

--loans
INSERT INTO loans ("loan_id", "general_id", "name", "phone_number", "email", "amount",
"rate", "payment", "term", "date")
VALUES (54603 ,58659 , 'Bank of the United States' ,44444444
,'unitedstatesbank456@yahoo.com' ,66868 ,50 ,13373.6,'monthly', '2020-08-07' );
INSERT INTO loans ("loan_id", "general_id", "name", "phone_number", "email", "amount",
"rate", "payment", "term", "date")
VALUES (35426 ,52895 ,'Bank of the United States' ,44444444
,'unitedstatesbank456@yahoo.com' ,8067 ,50 ,1613.4 ,'monthly' ,'2016-07-07' );
INSERT INTO loans ("loan_id", "general_id", "name", "phone_number", "email", "amount",
"rate", "payment", "term", "date")
VALUES (34550 ,96251 ,'Bank of the United States' ,44444444
,'unitedstatesbank456@yahoo.com' ,91690 ,50 ,18338 ,'monthly' ,'2016-07-07' );
INSERT INTO loans ("loan_id", "general_id", "name", "phone_number", "email", "amount",
"rate", "payment", "term", "date")
VALUES (76077 ,23353 ,'American Bank' ,44444444,'unitedstatesbank456@yahoo.com' ,60546
,50 ,12109.2 ,'monthly' ,'2016-07-07' );
INSERT INTO loans ("loan_id", "general_id", "name", "phone_number", "email", "amount",
"rate", "payment", "term", "date")
VALUES (46325 ,82711 ,'American Bank' ,44444444 ,'unitedstatesbank456@yahoo.com'
,18890 ,50 ,3778 ,'monthly' ,'2016-07-07' );
INSERT INTO loans ("loan_id", "general_id", "name", "phone_number", "email", "amount",
"rate", "payment", "term", "date")
VALUES (26360 ,45782 ,'American Bank' ,44444444 ,'unitedstatesbank456@yahoo.com'
,92961 ,50 ,18592.2 ,'yearly' ,'2016-07-07' );

--pro_forma
INSERT INTO pro_forma ("proforma_id", "prospective_id","projected_id", "epu_id", "mix_id",
"expenses_seller_proforma_id", "prior_id", "noi_id", "roll_id", "actual_income",
"future_potential_income", "actual_potential_income")
VALUES (60504 ,43906 ,62895 ,72945 ,55404 ,44522 ,67686 ,41091 ,80868 ,904470536
,107037685 ,287425317 );
```

```
INSERT INTO pro_forma ("proforma_id", "prospective_id","projected_id", "epu_id", "mix_id",
"expenses_seller_proforma_id", "prior_id", "noi_id", "roll_id", "actual_income",
"future_potential_income", "actual_potential_income")
VALUES (66609 ,64317 ,39473 ,50394 ,63084 ,25010 ,12019 ,41091 ,52920 ,64859190
,115094487 ,37967025 );
INSERT INTO pro_forma ("proforma_id", "prospective_id","projected_id", "epu_id", "mix_id",
"expenses_seller_proforma_id", "prior_id", "noi_id", "roll_id", "actual_income",
"future_potential_income", "actual_potential_income")
VALUES (20943 ,20030 ,80280 ,25475 ,79357 ,63882 ,40787 ,46306 ,28352 ,79935517
,764874522 ,91281479 );
INSERT INTO pro_forma ("proforma_id", "prospective_id","projected_id", "epu_id", "mix_id",
"expenses_seller_proforma_id", "prior_id", "noi_id", "roll_id", "actual_income",
"future_potential_income", "actual_potential_income")
VALUES (50695 ,60711 ,85795 ,28775 ,88556 ,29215 ,89921 ,27938 ,95873 ,13998581
,658454385 ,88857164 );
INSERT INTO pro_forma ("proforma_id", "prospective_id","projected_id", "epu_id", "mix_id",
"expenses_seller_proforma_id", "prior_id", "noi_id", "roll_id", "actual_income",
"future_potential_income", "actual_potential_income")
VALUES (74029 ,32679 ,75907 ,72129 ,24966 ,16078 ,89921 ,56197 ,31350 ,28410219
,568303484 ,3819889 );
INSERT INTO pro_forma ("proforma_id", "prospective_id","projected_id", "epu_id", "mix_id",
"expenses_seller_proforma_id", "prior_id", "noi_id", "roll_id", "actual_income",
"future_potential_income", "actual_potential_income")
VALUES (31522 ,32591 ,59847 ,58703 ,52557 ,36479 ,89921 ,23027 ,32136 ,54978242
,139083570 ,59204046 );
```

Views

```
/* In the case of this particular company, it is unnecessary that employees who
  are not general managers know information pertaining to a syndication as
  some of that information is sensitive and could cause detriment to the deal
  if the information gets leaked. The following querie creates a view called
  not_GM. It would create a view of the listed columns for employees who are
  not GM's. However, every employee at the moment is also a general manager. */

  CREATE VIEW not_GM AS
          SELECT synd_id, general_manager.general_id AS ge, status
     FROM syndication, general_manager, employee

/* The following view lists the loan attributes of all the loans that are given
  by the Bank of the United States. The general manager in charge of the loans
  at that bank do not need to see information pertaining to a different bank. */
```

```
CREATE VIEW bankOfTheUnitedStatesView AS
SELECT loan_id, general_id, name, phone_number, email, amount, rate, payment, term, date
FROM loans
WHERE name = 'Bank of the United States';


/* The following view creates a new employee table without the salary attribute.
  It is meant for employees who do not have jurisdiction over salaries and thus
  should not be able to see others' salaries.
*/

CREATE VIEW newEmployee AS
        SELECT employee_id, address_id, office_id, role, team, phone_number, email
        FROM employee;

/*
   This view combines all the attributes of the tables that would naturally be
   supplied by the seller of the prospective acquisition. This would be used by
   and outside party who should not have access to the data the investment
   compiled independly and added to the pro forma. (Not an updatable view)
 */
CREATE VIEW sellerProForma AS
SELECT rent_roll.type AS rt, unit_mix_and_rent_schedule.rent AS ur, --- the attributes from
different tables that share the same name had to be aliased
      rent_roll.rent AS rr,
      unit_mix_and_rent_schedule.type AS ut, appt_num, roll_id,
      "size_per_square", mix_id, num_units, square_feet, total_rent, "rent_per_square_foot",
      expenses_seller_proforma_id, repairs_maintence, utilities, taxes, insurance,
      reserve, total_expenses
FROM rent_roll, expenses_seller_proforma, unit_mix_and_rent_schedule
```

TRIGGERS

```
/*  Event: An employee's personal information needs to be changed.
   Action: All of the attributes are changed to new values.
*/


CREATE OR REPLACE FUNCTION employee_change()
  RETURNS trigger
  LANGUAGE plpgsql
  AS
```

```
   $$
    BEGIN
      INSERT INTO employee(employee_id, address_id, office_id, role, team, phone_number,
email, salary)
      VALUES (NEW.employee, NEW.address_id, NEW.office_id, NEW.role, NEW.team,
NEW.phone_number, NEW.email, NEW.salary);

      RETURN NEW;

    END;
   $$;

CREATE TRIGGER trg_employee_change
   AFTER INSERT ON general_manager
   FOR EACH ROW
   EXECUTE PROCEDURE employee_change();

/*  Event: An address needs to be changed or updated
   Action: All of the attributes are changed to new values.
*/


CREATE OR REPLACE FUNCTION change_address ()
RETURNS TRIGGER
LANGUAGE plpgsql
AS
$$
        BEGIN
                INSERT INTO addresses (address_id, address_line_one, address_line_two, city,
state, zip, country)
                VALUES (old.address_id, new.address_line_one, new.address_line_two,
new.city, new.state, new.zip, new.country);

                RETURN NEW;
        END;
$$;

CREATE TRIGGER trg_change_address
AFTER UPDATE ON employee
FOR EACH ROW EXECUTE PROCEDURE change_address();
```

Function

```
/*
   A function that calculates the capitalization rate which is the net
   operating income divded by the purchase price. This number helps determine
   the property's actual worth.
*/

CREATE OR REPLACE FUNCTION capitalization_rate(noi_id integer, prospective_id integer)
   RETURNS NUMERIC (11, 8)
   LANGUAGE plpgsql
   AS
   $$
    DECLARE
      cap NUMERIC (11, 8);
    BEGIN
       SELECT (noi_total / purchase_price) INTO cap
                  FROM pro_forma AS p JOIN net_operating_income AS n ON p.noi_id = n.noi_id
                        JOIN prospective_acquisitions AS a ON p.prospective_id =
a.prospective_id;
                  RETURN cap;
          END;
   $$;


SELECT capitalization_rate(41091, 43906); ---outputs 0.05881802


/*
  The following function is used to compute the Cash on Cash Profit. Cash on Cash
  Profit is equal to profit or income divided by down payment. Essentially, this determines
  the anual return that will be made for the respective property.
*/

CREATE OR REPLACE FUNCTION cash_on_cash(noi_id INTEGER, prospective_id
INTEGER)
  RETURNS NUMERIC (11, 8)
  LANGUAGE plpgsql
  AS
  $$
     DECLARE
       cashon NUMERIC (11, 8);
     BEGIN
        SELECT (income_total / down_payment) INTO cashon
        FROM pro_forma AS p JOIN prospective_acquisitions AS a ON
            p.prospective_id = a.prospective_id
```

```
            JOIN net_operating_income AS n ON n.noi_id = p.noi_id;
      RETURN cashon;
     END;
  $$;

SELECT cash_on_cash(41091, 43906); --- outputs 108.21709364
```

Procedure

```
/*
  A procedure that adds the late fee to a tenant's lease_agreement when they
  are late on rent.
*/

CREATE OR REPLACE PROCEDURE late_fee(foo integer)
          LANGUAGE plpgsql
          AS
          $$
                    BEGIN
                    UPDATE lease_agreement
                              SET rent = rent + fee_amount
                              WHERE date = current_date AND lease_id = foo;


    END;$$;

UPDATE lease_agreement
SET date = '2022-05-04'
WHERE lease_id = 63053

----the rent amount before calling the procedure
SELECT rent
FROM lease_agreement
WHERE lease_id = 63053 --- outputs: 90600.00


CALL late_fee(63053);

----the rent after the late fee was applied
SELECT rent
FROM lease_agreement
WHERE lease_id = 63053 ---outputs: 90670.00
```

DML

```
/* A tenant with id 53071 is moving out. The following statement
   deletes their tuple in the database.
*/

        DELETE FROM tenant
        WHERE tenant_id = 53071;
```

Queries

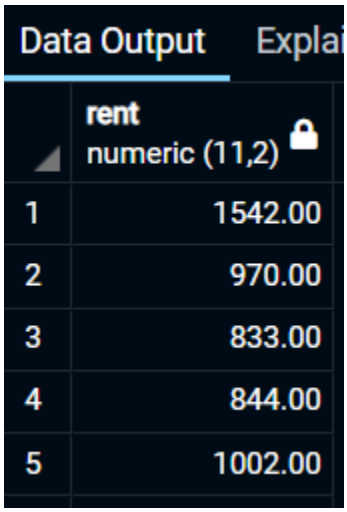
Q#1.

```
/*
  Q#1.
  There was an error in the leasing department where all the rents were 10 times
  the amount they should be. The following query updates all the lease agreements
  so that the rents are the correct amount.
*/

SELECT rent
FROM lease_agreement

UPDATE lease_agreement
SET rent = rent * 100.00
WHERE rent > 2000.00;

SELECT rent
FROM lease_agreement
```



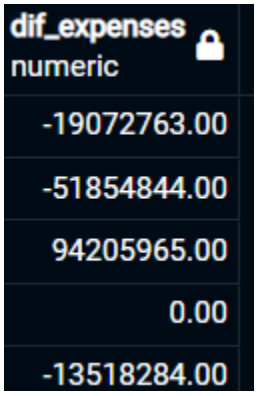
Data Output | Explai

	rent numeric (11,2)
1	1542.00
2	970.00
3	833.00
4	844.00
5	1002.00

Q#2.

```
/*
  Q#2.
  The following query finds the difference between the prior year actual total
  expenses and the total expenses reported in the pro forma by the seller.
*/

SELECT (expenses_prior_year_actuals.total_expenses -
expenses_seller_proforma.total_expenses) AS dif_expenses
FROM expenses_prior_year_actuals, expenses_seller_proforma, pro_forma
WHERE expenses_prior_year_actuals.prior_id = pro_forma.prior_id
        AND pro_forma.expenses_seller_proforma_id =
expenses_seller_proforma.expenses_seller_proforma_id;
```



dif_expenses numeric
-19072763.00
-51854844.00
94205965.00
0.00
-13518284.00

Q#3.

```
/*
  Q#3
  Since, Kim and Kanye broke up, Kanye is moving out. The following query
  deletes Kanye's record from the tenant table.
*/

DELETE FROM tenant
WHERE name = 'Kanye West';

 --- A SET OF UPDATE STATEMENTS TO FIX THE address_id'S IN THE current_properties
table
 UPDATE current_properties
SET address_id = 97301
WHERE  property_id = 85610;

UPDATE current_properties
```

```
SET address_id = 57189
WHERE  property_id = 68262;

UPDATE current_properties
SET address_id = 60033
WHERE  property_id = 35574;

UPDATE current_properties
SET address_id = 44069
WHERE  property_id = 97587;

UPDATE current_properties
SET address_id = 82520
WHERE  property_id = 98385;

UPDATE current_properties
SET address_id = 33062
WHERE  property_id = 86534;

UPDATE current_properties
SET address_id = 70498
WHERE  property_id = 86268;

UPDATE current_properties
SET address_id = 35951
WHERE  property_id = 40299;

UPDATE current_properties
SET address_id = 57423
WHERE  property_id = 93292;
```

Q#4.

```
/*
  Q#4.
  The following query finds the id and name of all the tenants living on a
  property in California.
*/

SELECT tenant_id, name
FROM addresses AS a JOIN current_properties AS c ON a.address_id = c.address_id
    JOIN tenant AS t ON t.property_id = c.property_id
WHERE state = 'CA';
```

	tenant_id [PK] integer	name character varying (50)
1	12157	Kim Kardashian
2	57234	Kayne West
3	37217	Marget Thatcher
4	27031	George Washington
5	75874	Ghandi

Q#5.

```
/*
        Q#5.
  The following query returns a list of the names of employee_id who are not
  general managers.
*/

SELECT DISTINCT employee.employee_id
FROM employee, general_manager
WHERE employee.employee_id NOT IN (SELECT employee.employee_id
                                   FROM employee, general_manager
                                   WHERE employee.employee_id =
general_manager.employee_id);
```

	employee_id [PK] integer

Q#6.

```
*
        Q#6.
        The following query compares the noi_total reported in the net_operating_income
   and the difference between the income_total and expense_total, and gives the
   property_id for the prospective property.
*/

SELECT c.prospective_id, noi_total, (income_total - expense_total) as dif_noi
FROM net_operating_income AS n JOIN pro_forma AS pf ON n.noi_id = pf.noi_id
         JOIN prospective_acquisitions AS c ON pf.prospective_id = c.prospective_id;
```

	prospective_id integer	noi_total numeric (11,2)	dif_noi numeric
1	43906	9816653.00	9816653.00
2	64317	9816653.00	9816653.00
3	20030	9490678.00	9490678.00
4	32679	199268276.00	199268276.00
5	32591	111180812.00	111180812.00

Q#7.

```
/*
        Q#7.
  The following query gives the employee ids of the employees who are not
  landlords using the except clause.
*/

(SELECT employee_id
FROM employee)
EXCEPT
(SELECT e.employee_id
FROM employee AS e JOIN landlord AS l ON e.employee_id = l.employee_id);
```

	employee_id integer

Q#8.

```
/*
        Q#8.
  The following query gives the rent averages per unit type according
  unit mix and rent schedule.
*/

SELECT type, ROUND(AVG(rent), 2) AS rent_avg
FROM unit_mix_and_rent_schedule
GROUP BY type;
```

	type character varying (50)	rent_avg numeric
1	1/1	468.00
2	3/2	803.33
3	2/1	1063.00

Q#9.

```
/*
        Q#9.
  The following query finds the occupancy rates for current properties by
  dividing the total occupants by the number of units for properties in
  California. It also gives the corresponding street address listed in
   alphabetical order.
*/

SELECT address_line_one, (num_occupants / num_units) AS occupancy_rate
FROM addresses AS a JOIN current_properties AS c ON c.address_id = a.address_id
WHERE state = 'CA'
ORDER BY address_line_one ASC;
```

	address_line_one character varying (50)	occupancy_rate numeric
1	1180 S. Beverly Drive	0.44380403458213256484
2	123 Rodeo Drive	21.7272727272727273
3	132 Rodeo Drive	1.1268656716417910
4	1828 Courtney Avenue	10.7976190476190476
5	433 Forman Ave	2.3684210526315789
6	8033 Sunset Boulevard	2.7945619335347432
7	955 S. Carrillo Drive	0.20000000000000000000

Application Code

In [1]:
```
pip install psycopg2-binary
```

```
Requirement already satisfied: psycopg2-binary in c:\users\f\anaconda
3\lib\site-packages (2.9.3)
Note: you may need to restart the kernel to use updated packages.
```

The following code conntects to the database, executes a basic query, and displays the result to the user.

In [2]:
```
import psycopg2

##connecting to the database
connection = psycopg2.connect(user="postgres", password="Laxislife15!"
##Creating a cursor
cursor = connection.cursor()
cursor.execute(f"SELECT * FROM current_properties")
query_1 = cursor.fetchall()
print(query_1)
cursor.close()
connection.close()
```

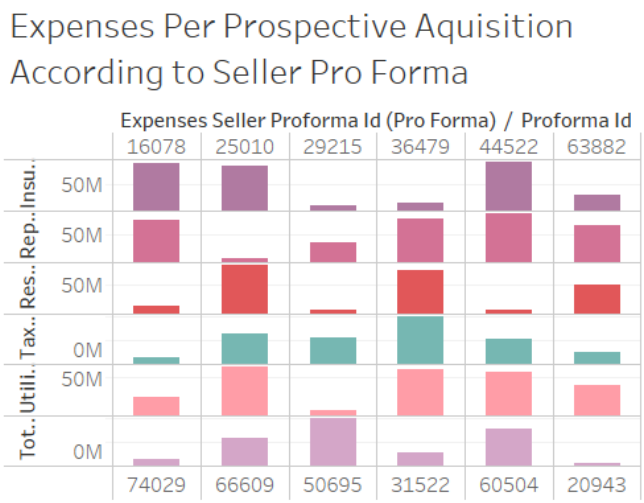
```
[(85610, 97301, Decimal('408.00'), datetime.date(2000, 12, 31), Decim
al('150000.00'), Decimal('2052033.00'), 638), (68262, 57189, Decimal
('402.00'), datetime.date(2018, 6, 6), Decimal('9361602.00'), Decimal
('6093446.00'), 453), (35574, 60033, Decimal('331.00'), datetime.date
(2020, 6, 4), Decimal('6269812.00'), Decimal('4491732.00'), 925), (97
587, 44069, Decimal('160.00'), datetime.date(2020, 6, 4), Decimal('51
49029.00'), Decimal('9631141.00'), 32), (98385, 82520, Decimal('22.0
0'), datetime.date(2020, 6, 4), Decimal('4803640.00'), Decimal('82418
35.00'), 478), (86534, 33062, Decimal('133.00'), datetime.date(2020,
6, 4), Decimal('570802.00'), Decimal('1553028.00'), 315), (86268, 704
```

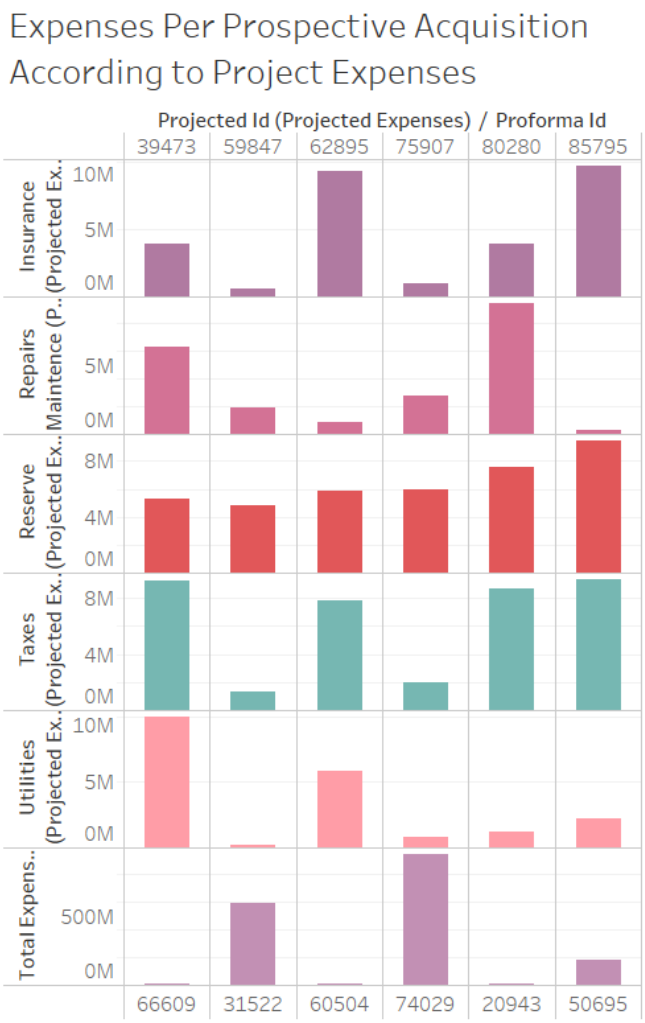
The following python code accesses the stored procedure, late_fee, and utilizes a try and except clause.

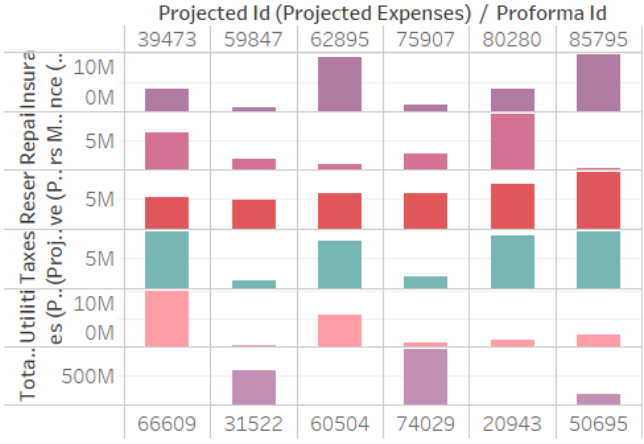
In [20]:

```
import psycopg2
def bar(foo):
    conn = None
    try:
        params = config()
        conn = psycopg2.connect(**params)
        cur = conn.cursor()
        cur.execute('CALL late_fee(%s)', (foo))
        conn.commit()
        cur.close()
    except (Exception, psycopg2.DatabaseError) as error:
        print(error)
    finally:
        if conn is not None:
            conn.close()
```

Tableau Dashboard







*** Please let me know if you need any more information regarding Tableau and how I used it and connected it to my database.