Based on your database schema for **Real Estate Management System** Database, here's a suggested report format. This assumes the report is for someone wanting to understand the state of the real estate business represented by the database.

I. Data Description

* Describe the tables in the database and their relationships.

  * "The Real_Estate_DB database consists of five tables:

    Property: Stores information about properties (address, type, size, rent).

    Tenant: Stores information about tenants (name, contact, lease start/end dates).

    Lease Agreement: Links properties and tenants, and stores lease terms.

    Maintenance Request: Tracks maintenance requests.

    Payment: Records payment transactions related to leases."

  * Include an Entity-Relationship Diagram (ERD) if possible.
  * Describe the data types of the columns in each table.
  * Describe the Constraints of the columns in each table.

II. Property Analysis

* Analize the data in the `Property` table. Include:
* Distribution of property types (Residential vs. Commercial).
* Descriptive statistics of property sizes (mean, median, range). (Use a table)
* Descriptive statistics of rent amounts. (Use a table)
* Identify properties with the highest and lowest rent.
* Geographic distribution of properties (if the Address field allows for it, e.g., by city).

III. Tenant Analysis

* Analize the data in the `Tenant` table. Include:
* Number of tenants.
* Statistics on lease start and end dates (e.g., average lease duration).
* Any other relevant analysis of tenant data.

IV. Lease Agreement Analysis

* Analize the data in the `Lease Agreement` table. Include:

* Number of lease agreements.

* Relationship between property type and monthly rent.

* Distribution of lease start and end dates.

* Analysis of the stored procedure `Get Active Leases()` .

* Average monthly rent.

* Identify any correlations between property size and monthly rent.

V. Financial Analysis (Payment)

* Analize the data in the `Payment` table. Include:

* Total payments recorded.

* Distribution of payment methods (Cash, Check, etc.).

* Average payment amount.

* Analysis of the stored procedure `Get Payments By Lease()`.

* Trends in payment amounts over time. (Use a chart)

VI. Maintenance Request Analysis

* Analize the data in the `Maintenance Request` table. Include:

* Total number of maintenance requests.

* Distribution of maintenance request statuses (Pending, In Progress, Completed).

* Most common types of maintenance requests (if the Description field allows for categorization).

* Average time to complete maintenance requests (if the data allows). * Number of requests per property.

Here's an overview of the stored procedures you've created, based on the report format and your database schema:

Stored Procedure Overview

Here's a summary of the stored procedures defined for the Real_Estate_DB database:

Get Property Rent Rankings():

- Purpose: Retrieves property data with rent rankings.
- Parameters: None.
- Logic: Uses RANK(), DENSE_RANK(), and ROW_NUMBER() window functions to rank properties based on their RentAmount.
- Use in Real Estate Business: Useful for identifying high-value properties, comparing rental prices, and marketing properties based on their ranking.
- Efficiency: Efficient for relatively small datasets. For very large datasets, consider adding an index on RentAmount if performance is critical.

Add Property():

- Purpose: Adds a new property to the Property table.
- Parameters: p_Address (VARCHAR), p_Type (ENUM), p_Size (DECIMAL), p_RentAmount (DECIMAL).
- Logic: Inserts a new row into the Property table with the provided property details.
- Use in Real Estate Business: Essential for adding new properties to the database as the business acquires them.
- Efficiency: Efficient for single-row inserts.

Add Tenant():

- Purpose: Adds a new tenant to the Tenant table.
- Parameters: p_Name (VARCHAR), p_ContactInfo (VARCHAR), p_LeaseStartDate (DATE), p_LeaseEndDate (DATE).
- Logic: Inserts a new row into the Tenant table with the provided tenant details.
- Use in Real Estate Business: Essential for recording new tenants as they sign leases.
- Efficiency: Efficient for single-row inserts.

Create Lease Agreement():

- Purpose: Creates a new lease agreement in the LeaseAgreement table.
- Parameters: p_PropertyID (INT), p_TenantID (INT), p_StartDate (DATE), p_EndDate (DATE), p_MonthlyRent (DECIMAL).
- Logic: Inserts a new row into the LeaseAgreement table, establishing a link between a property and a tenant with specific lease terms.
- Use in Real Estate Business: Crucial for recording the terms of a lease when a tenant agrees to rent a property.
- Efficiency: Efficient for single-row inserts. The foreign key constraints (fk_property, fk_tenant) ensure data integrity.

Add Payment():

- Purpose: Records a payment made for a lease.
- Parameters: p_LeaseID (INT), p_Amount (DECIMAL), p_Date (DATE), p_Method (ENUM).
- Logic: Inserts a new row into the Payment table, linking the payment to a specific lease agreement.
- Use in Real Estate Business: Essential for tracking financial transactions and managing revenue.
- Efficiency: Efficient for single-row inserts. The foreign key constraint (fk_payment_lease) ensures data integrity.

Submit Maintenance Request():

- Purpose: Records a new maintenance request.
- Parameters: p_PropertyID (INT), p_TenantID (INT), p_Description (TEXT), p_DateSubmitted (DATE).
- Logic: Inserts a new row into the MaintenanceRequest table, linking the request to a property and tenant.
- Use in Real Estate Business: Important for managing property maintenance and responding to tenant needs.Efficiency: Efficient for single-row inserts. The foreign key constraints (fk_maintenance_property, fk_maintenance_tenant) ensure data integrity.

Get Lease By Tenant():

- Purpose: Retrieves lease details for a specific tenant.
- Parameters: P-Tenant ID (INT).
- Logic: Joins the Lease Agreement and Property tables to retrieve lease information based on the provided Tenant ID.
- Use in Real Estate Business: Useful for retrieving lease information for a specific tenant, e.g., when a tenant inquires about their lease.
- Efficiency: The efficiency depends on the indexing of the Tenant ID column in the Lease Agreement table. Adding an index would improve performance.

Get Payments By Lease():

- Purpose: Retrieves all payments for a specific lease.
- Parameters: P-Lease ID (INT).
- Logic: Selects all rows from the Payment table that match the provided Lease ID.
- Use in Real Estate Business: Useful for financial tracking, such as verifying payments for a specific lease or generating payment history.
- Efficiency: The efficiency depends on the indexing of the Lease ID column in the Payment table. Adding an index would improve performance.

Update Payment Status():

- Purpose: Updates the amount of a specific payment.
- Parameters: P_PaymentID (INT), P_NewAmount (DECIMAL).
- Logic: Updates the Amount column in the Payment table for the specified PaymentID.
- Use in Real Estate Business: Useful for correcting payment amounts.
- Efficiency: Efficient for single-row updates. An index on Payment ID would be beneficial.

Get Active Leases():

- Purpose: Retrieves all currently active leases.
- Parameters: None.
- Logic: Joins the Lease Agreement, Property, and Tenant tables and filters for leases where the End Date is greater than or equal to the current date.
- Use in Real Estate Business: Essential for getting an overview of current lease obligations, managing occupancy, and identifying upcoming lease expirations.
- Efficiency: The efficiency depends on the indexing of the End Date column in the Lease Agreement table. Adding an index would improve performance.

Delete Lease():

- Purpose: Deletes a lease agreement.
- Parameters: p_Lease ID (INT).
- Logic: Deletes the row from the Lease Agreement table with the specified LeaseID.
- Use in Real Estate Business: Used to remove a lease agreement from the database, for example, when a lease is terminated.
- Efficiency: Efficient for single-row deletions. An index on LeaseID would be beneficial.

Occupancy Trend():

- Purpose: Analize monthly occupancy trends.
- Parameters: None.
- Logic: Calculates the number of occupied units per month and year, and the month-over-month change in occupancy.
- Use in Real Estate Business: Provides insights into how the occupancy changes over time.
- Efficiency: Efficient, but can be improved with proper indexing on the StartDate column of the LeaseAgreementtable.