Real Estate Data Analysis (SQL)

Project Overview:

In today’s dynamic property market, data holds the key to smarter investments. This project uncovers the hidden patterns behind real estate sales using the power of **SQL**-driven analytics. By transforming raw property transactions into clear, actionable insights, it sheds light on sales performance, agent productivity, client behavior, and market trends across regions and time periods; helping decision-makers make smarter, evidence-based investment choices.

SQL

Q1 — How have real estate sales and customer activity evolved yearly?

SQL Script:

```
-- Change-Over-Time Trends
SELECT
  DATE_FORMAT(sale_date, '%Y') AS yearly_sales,
  FORMAT(SUM(sale_price), 0) AS total_sales_revenue,
  COUNT(DISTINCT sales_id) AS total_number_of_sales,
  COUNT(DISTINCT client_id) AS total_clients
FROM gold.fact_sales
GROUP BY yearly_sales
ORDER BY yearly_sales;
```

Output Snapshot:

year	total_sales_revenue	total_number_of_sales	total_clients
2018	634,354,928	573	506
2019	612,710,680	555	487
2020	682,156,448	624	544
2021	635,239,014	602	531
2022	591,643,475	559	505
2023	599,784,003	572	498
2024	599,806,751	577	510

Analysis:

Real estate sales and client activity peaked in 2020, reflecting strong market demand. Since then, sales have stabilized with minor fluctuations, and 2024 shows slightly more transactions but flat revenue, suggesting smaller average deal sizes. The trend highlights the need to target marketing efforts and refine product offerings to increase revenue while maintaining client engagement.

Q2 — What is the monthly trend in total and average sales, and how do they accumulate over time?

SQL Script:

```
-- Monthly Sales & Cumulative Totals
SELECT
  month,
  total_sales_revenue,
  total_sales_average,
  SUM(total_sales_revenue) OVER(ORDER BY monthly_sales) AS running_total_sales,
  AVG(total_sales_average) OVER(ORDER BY monthly_sales) AS running_average_sales
FROM (
  SELECT
    DATE_FORMAT(sale_date, '%m') month,
    SUM(sale_price) AS total_sales_revenue,
    AVG(sale_price) AS total_sales_average
  FROM gold.fact_sales
  GROUP BY monthly_sales
) t;
```

Output Snapshot:

month	total_sales_revenue	total_sales_average	running_total_sales	running_avg_sales
01	343,411,482	1,053,409	343,411,482	1,053,409
02	352,206,832	1,042,032	695,618,314	1,047,720
03	367,863,537	1,091,583	1,063,481,851	1,062,342
04	394,926,326	1,093,978	1,458,408,177	1,070,251
12	350,427,250	1,058,693	4,355,695,299	1,072,281

Analysis:

Monthly sales show a steady cumulative growth reaching over $4.35B by December, with notable peaks in June and October, indicating seasonal buying patterns. The average sale price remains stable, suggesting consistent property valuation across the year. This insight can guide inventory planning, seasonal marketing campaigns, and financial forecasting to maximize revenue capture during high-demand months.

Q3 — Analyze the yearly perfomance of property types by comparing each property type to both its average sales perfomance and the previous years sales

SQL Script:

```
-- Property Type Performance Analysis
WITH yearly_property_sales AS (
  SELECT
    DATE_FORMAT(f.sale_date, '%Y') year,
    p.property_type,
    SUM(f.sale_price) AS current_sales
  FROM gold.fact_sales f
  LEFT JOIN gold.dim_properties p ON f.property_id = p.property_id
  GROUP BY DATE_FORMAT(f.sale_date, '%Y'), p.property_type
)
SELECT
  year,
  property_type,
  current_sales,
  AVG(current_sales) OVER(PARTITION BY property_type) AS average_sales,
  current_sales - AVG(current_sales) OVER(PARTITION BY property_type) AS diff_avg,
  CASE
    WHEN current_sales - AVG(current_sales) OVER(PARTITION BY property_type) > 0
    THEN 'Above Average'
    ELSE 'Below Average'
  END AS average_change,
  LAG(current_sales) OVER(PARTITION BY property_type ORDER BY year) AS previous_sales,
  current_sales - LAG(current_sales) OVER(PARTITION BY property_type ORDER BY year) AS diff_py,
  CASE
    WHEN current_sales - LAG(current_sales) OVER(PARTITION BY property_type ORDER BY year) > 0
    THEN 'Increase'
    ELSE 'Decrease'
  END AS diff_py_label
FROM yearly_property_sales
ORDER BY year, property_type;
```

Output Snapshot:

year	property_typ	current_sale	average_sal	avg_chang	prevoius_sal	diff_py
2018	Commercial	315,756,164	313,133,818	Above Avg	—	No Change
2018	Residential	318,598,764	309,108,368	Above Avg	—	No Change
2019	Commercial	277,817,149	313,133,818	Below Avg	315,756,164	Decrease
2019	Residential	334,893,531	309,108,368	Above Avg	318,598,764	Increase

Analysis:

Commercial properties consistently perform at or above average, providing stable revenue streams, while residential properties are more volatile, showing declines in several years post-2020. The trend suggests that commercial real estate is the backbone of sales stability, whereas residential requires careful pricing and promotional strategies to mitigate revenue fluctuations.

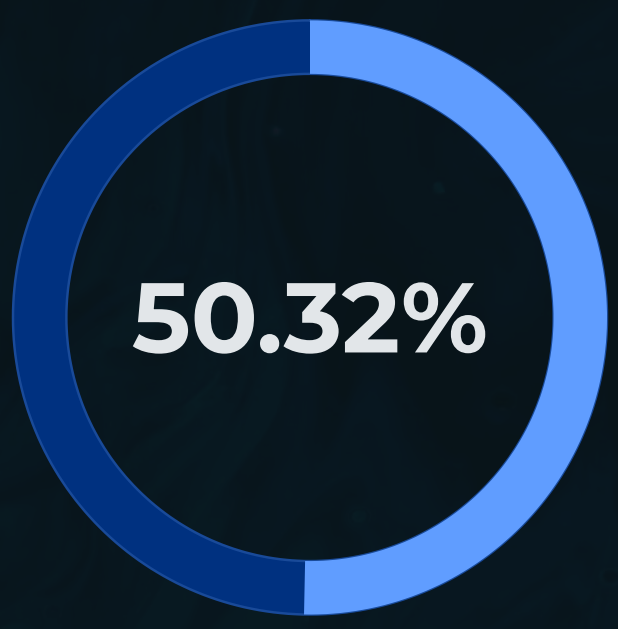
Q4 — Which property type contributes most to overall revenue?

SQL Script:

```
-- Q4: Part-to-Whole Revenue Contribution
WITH property_sales AS (
  SELECT
    p.property_type,
    SUM(f.sale_price) AS total_sales
  FROM gold.fact_sales f
  LEFT JOIN gold.dim_properties p
    ON f.property_id = p.property_id
  GROUP BY p.property_type
)
SELECT
  property_type,
  total_sales,
  SUM(total_sales) OVER() AS overall_sales,
  CONCAT(ROUND(total_sales/SUM(total_sales)
    OVER()*100,2),'%') AS percentage_of_total
FROM property_sales;
```

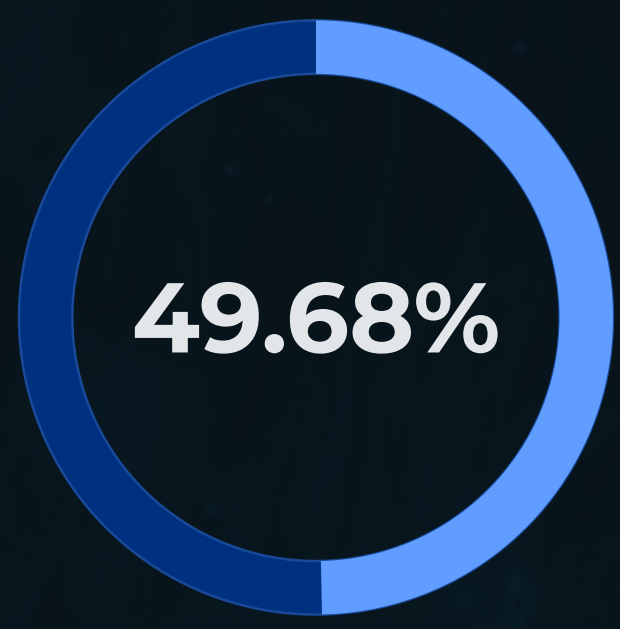
Output Snapshot:

Property_type	total_sales	% of total_sales
Commercial	2,191,936,724	50.32%
Residential	2,163,758,575	49.68%



Commercial

$2,191,936,724



Residential

$2,163,758,575

Analysis:

Both commercial and residential segments contribute nearly equally to overall revenue, with commercial slightly ahead. This indicates that balanced attention to both sectors is critical for maintaining total sales volume, while incremental gains in commercial performance could provide a marginal but meaningful revenue boost.

Q5 — How are property prices distributed, and which ranges capture the majority of transactions?

SQL Script:

```
-- Property Price Segmentation
WITH property_segments AS (
  SELECT
    f.property_id,
    p.property_type,
    f.sale_price,
    CASE
      WHEN sale_price < 500000 THEN 'Below 500K'
      WHEN sale_price BETWEEN 500000 AND 1000000 THEN '500K-1M'
      ELSE 'Above 1M'
    END AS cost_ranges
  FROM gold.fact_sales f
  LEFT JOIN gold.dim_properties p ON f.property_id = p.property_id
)
SELECT
  cost_ranges,
  COUNT(cost_ranges) AS total_properties
FROM property_segments
GROUP BY cost_ranges;
```

Output Snapshot:

2,161	1,093	808
Above 1M	**500K–1M**	**Below 500K**
Premium segment properties	Mid-tier properties	Entry-level properties

Analysis:

The majority of sales are in the premium segment above 1M, reflecting a market dominated by high-value properties. Mid-tier and lower-priced properties are less common, indicating potential growth opportunities for targeting these segments to expand market reach and diversify revenue sources.