PROJECT

ECOMMERCE SQL ANALYSIS

Prepared By :

Harsh Verma

Overview:

This project involves the design and implementation of a Retail Sales Data Warehouse using a star schema. It is developed to support efficient, fast, and scalable analysis of retail sales data.

Use Cases:

Sales trend analysis across different regions and time periods.

Customer behavior tracking.

Store and product performance evaluation.

Payment method performance and bank-wise analysis.

Objective:

To enable businesses to gain deep insights into their sales transactions across various dimensions like:

Time (date, month, year)

Customers (demographics, identifiers)

Items (products, pricing)

Stores (locations)

Payment methods (bank, transaction types)

```
--Create TABLE Transaction--
CREATE TABLE Trans_dim (
    id SERIAL PRIMARY KEY,
    payment_key CHAR(10) UNIQUE,
    trans_type VARCHAR(50),
    bank_name VARCHAR(100)
);

--  Create Store Dimension Table--
CREATE TABLE store_dim (
    id SERIAL PRIMARY KEY,
    store_key CHAR(10) UNIQUE,
    division VARCHAR(50),
    district VARCHAR(50),
    upazila VARCHAR(50)
);

--  Create Time Dimension Table--
CREATE TABLE time_dim (
    id SERIAL PRIMARY KEY,
    time_key CHAR(20) UNIQUE,
    date DATE,
    hour TIME,
    day INT,
    week CHAR(20),
    month CHAR(20),
    quarter CHAR(20),
    year INT
```

```
);
```

-- Create Item Dimension Table--

```
CREATE TABLE item_dim (
    id SERIAL PRIMARY KEY,
    item_key CHAR(20) UNIQUE,
    item_name VARCHAR(100),
    description TEXT,
    unit_price DECIMAL(10, 2),
    man_country VARCHAR(50),
    supplier VARCHAR(100),
    unit VARCHAR(20)
);
```

-- Create Customer Dimension Table--

```
CREATE TABLE customer_dim (
    id SERIAL PRIMARY KEY,
    customer_key CHAR(20) UNIQUE,
    name VARCHAR(100),
    contact_no BIGINT,
    nid BIGINT
);
```

-- Create Fact Table--

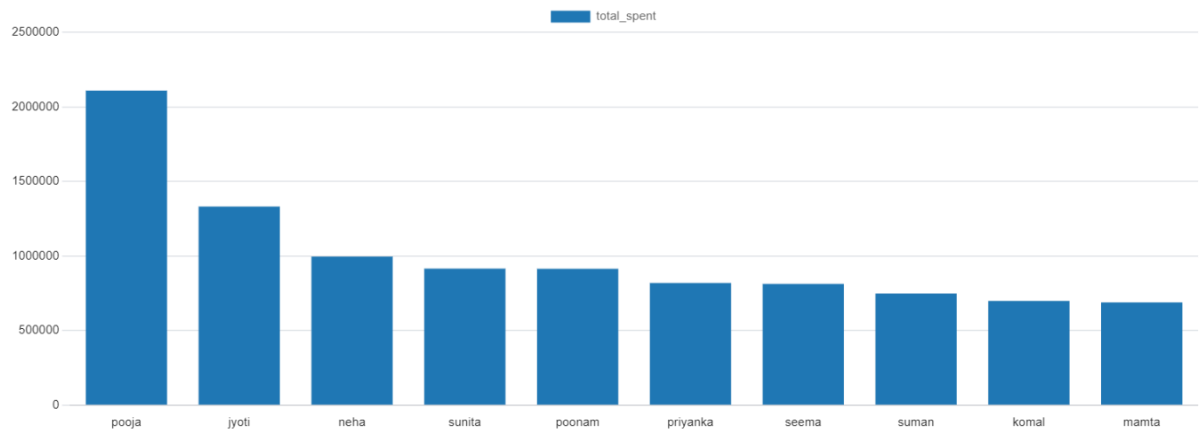
```
CREATE TABLE fact_table (
    id SERIAL PRIMARY KEY,
    payment_key CHAR(20),
    customer_key CHAR(20),
```

```
    time_key CHAR(20),
    item_key CHAR(20),
    store_key CHAR(20),
    quantity INT,
    unit VARCHAR(20),
    unit_price DECIMAL(10, 2),
    total_price DECIMAL(12, 2)
);
```

--Top 10 Customers by Spending--

```
WITH cte AS (
    SELECT
        customer_key,
        quantity,
        unit_price,
        quantity * unit_price AS total_value
    FROM fact_table
)
SELECT
    c.name AS customer_name,
    SUM(ct.total_value) AS total_spent
FROM cte ct
JOIN customer_dim c ON ct.customer_key = c.customer_key
GROUP BY c.name
ORDER BY total_spent DESC
LIMIT 10;
```

	customer_name character varying (100)	total_spent numeric
1	pooja	2109800.75
2	jyoti	1331696.25
3	neha	996121.00
4	sunita	915543.75



--Top 10 Items by Revenue--

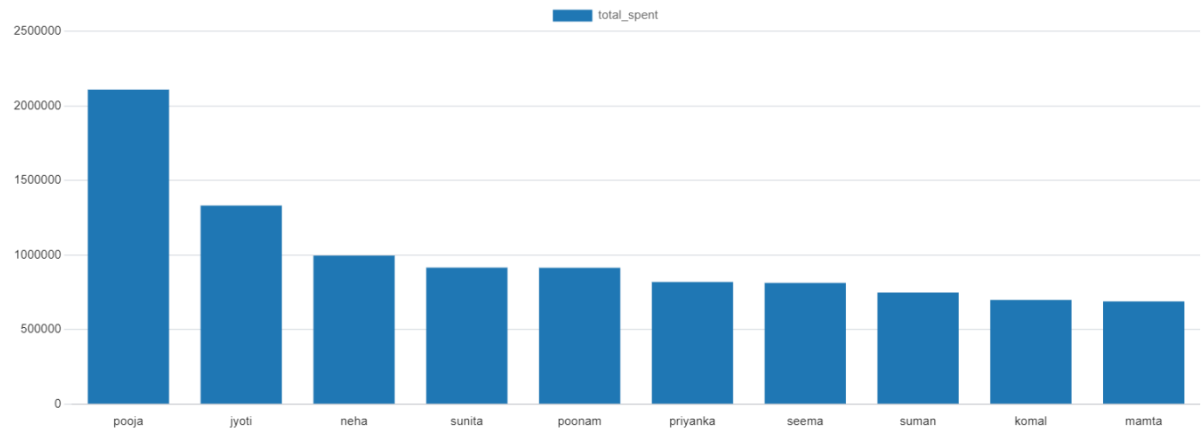
```
WITH cte AS (
    SELECT
        item_key,
        quantity,
        unit_price,
        quantity * unit_price AS total_value
    FROM fact_table
)
SELECT
    i.item_name,
    SUM(ct.total_value) AS total_revenue
FROM cte ct
JOIN item_dim i ON ct.item_key = i.item_key
GROUP BY i.item_name
ORDER BY total_revenue DESC
LIMIT 10;
```

	item_name character varying (100)	total_revenue numeric
1	Red Bull 12oz	1305700.00
2	K Cups Daily Chef Columbian Supremo	1245394.00
3	K Cups Original Donut Shop Med. Roast	1188843.00
4	K Cups Dunkin Donuts Medium Roast	1109760.00

--Monthly Sales (Any 5 Months)--

```
SELECT
    t.month,
    SUM(f.total_price) AS total_sales
FROM fact_table f
JOIN time_dim t ON f.time_key = t.time_key
GROUP BY t.month
ORDER BY t.month
LIMIT 5;
```

	month character (20)	total_sales numeric
1	Apr	8504634.75
2	Aug	8929475.25
3	Dec	8923480.50
4	Feb	8073877.50



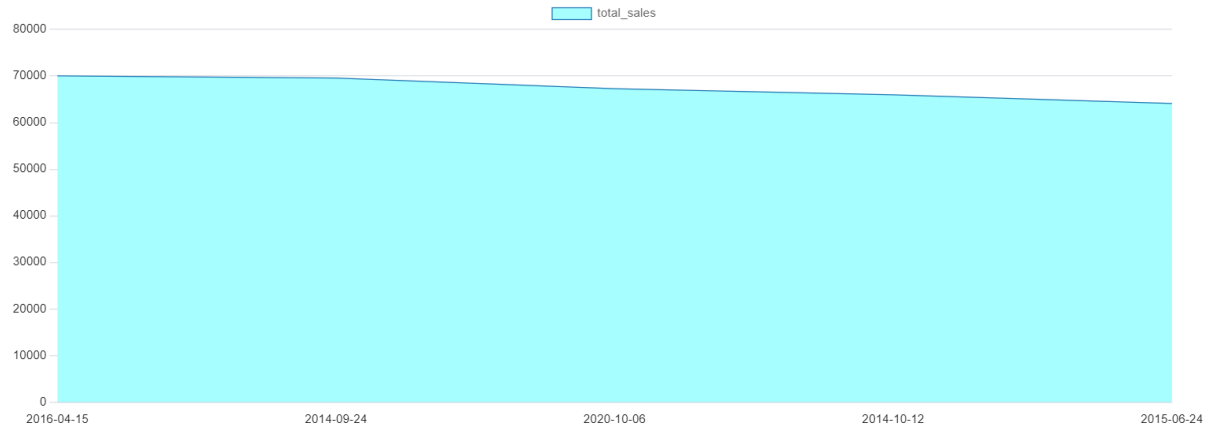
-- Top 5 Dates by Sales--

```
SELECT
    t.date,
    SUM(f.total_price) AS total_sales
FROM fact_table f
JOIN time_dim t ON f.time_key =
t.time_key
GROUP BY t.date
```

	date date	total_sales numeric
1	2016-04-15	70039.50
2	2014-09-24	69573.25
3	2020-10-06	67297.00
4	2014-10-12	65979.75

ORDER BY total_sales DESC

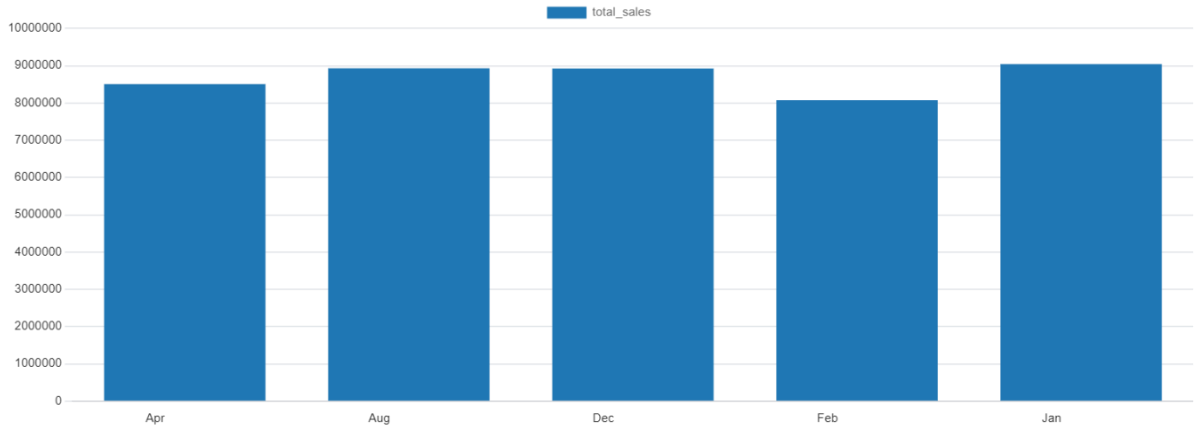
LIMIT 5;



--TOP 5 MONTHS—

```
SELECT
    t.month,
    SUM(f.total_price) AS total_sales
FROM fact_table f
JOIN time_dim t ON f.time_key = t.time_key
GROUP BY t.month
ORDER BY total_sales DESC
LIMIT 5;
```

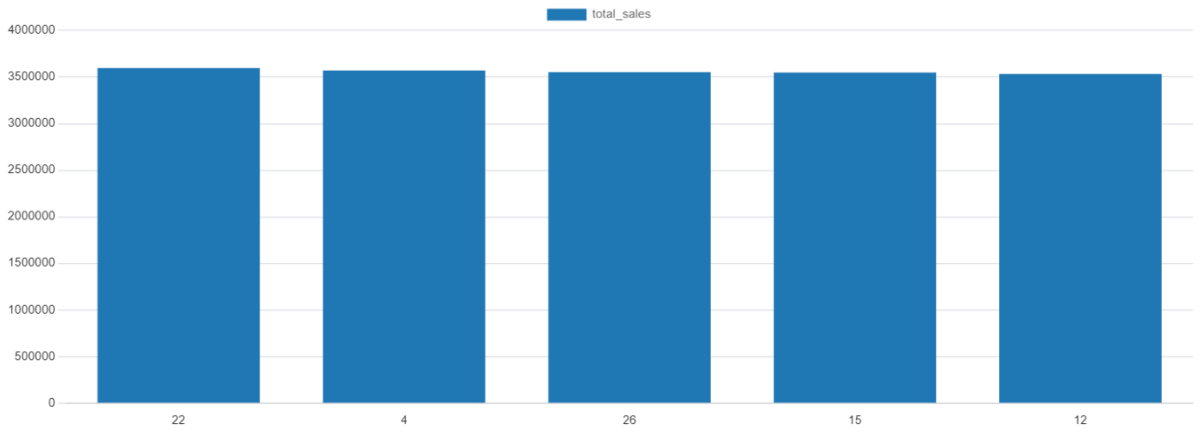
	month character (20)	total_sales numeric
1	May	9078002.50
2	Jul	9046580.00
3	Jan	9042244.00
4	Mar	8939152.00



--TOP 5 DAY--

```
SELECT
    t.day,
    SUM(f.total_price) AS total_sales
FROM fact_table f
JOIN time_dim t ON f.time_key = t.time_key
GROUP BY t.day
ORDER BY total_sales DESC
LIMIT 5;
```

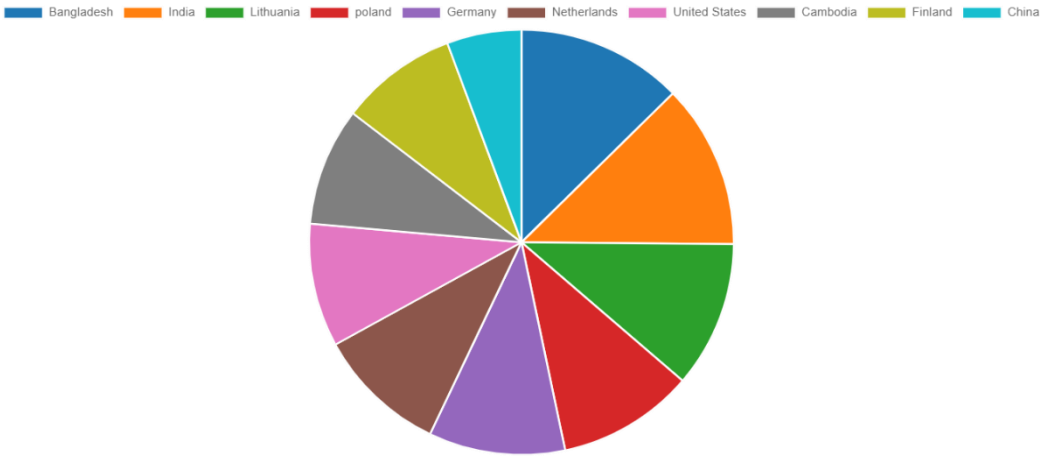
	day integer	total_sales numeric
1	22	3595900.25
2	4	3569375.00
3	26	3552391.75
4	15	3547880.25



--Revenue by Manufacturer Country--

```
SELECT
i.man_country,
    SUM(f.total_price) AS total_revenue
FROM fact_table f
JOIN item_dim i ON f.item_key = i.item_key
GROUP BY i.man_country
ORDER BY total_revenue DESC;
```

	man_country character varying (50)	total_revenue numeric
1	Bangladesh	13337300.50
2	India	13159323.25
3	Lithuania	11746920.00
4	poland	10976287.50

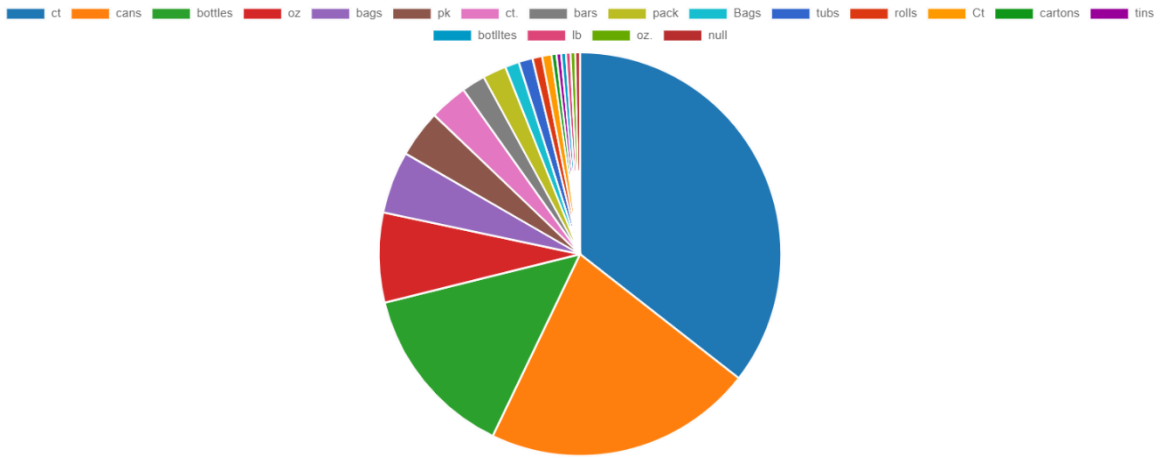


--Units Sold by Type--

```
SELECT
    f.unit,
    COUNT(*) AS transaction_count,
    SUM(f.quantity) AS total_units_sold
FROM fact_table f
GROUP BY f.unit
```

	unit character varying (20)	transaction_count bigint	total_units_sold bigint
1	ct	355938	2132313
2	cans	215602	1296045
3	bottles	140073	839599
4	oz	71956	431828

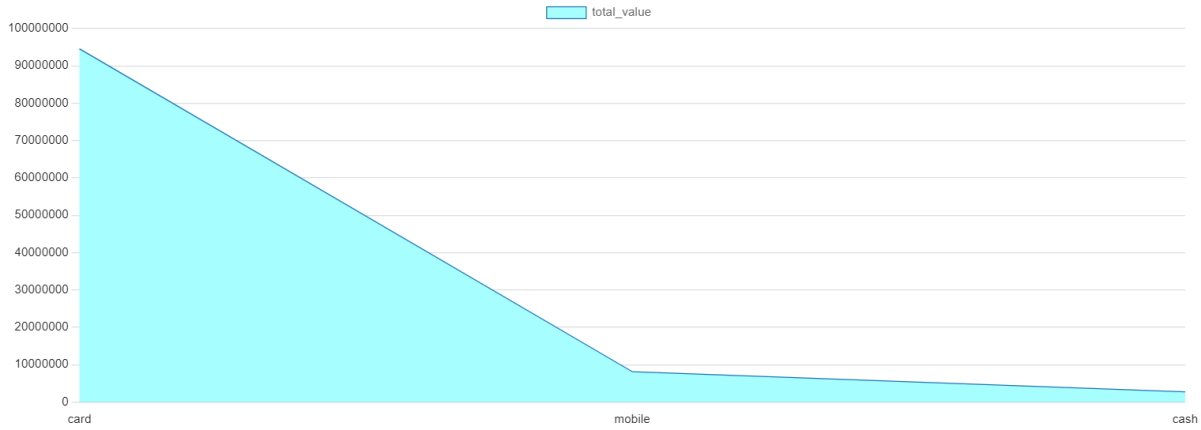
ORDER BY total_units_sold DESC;



--Top 10 Payment Types by Revenue--

```
SELECT
    td.trans_type,
    SUM(ft.quantity * ft.unit_price) AS total_value
FROM fact_table ft
JOIN trans_dim td ON ft.payment_key = td.payment_key
GROUP BY td.trans_type
ORDER BY total_value DESC
LIMIT 10;
```

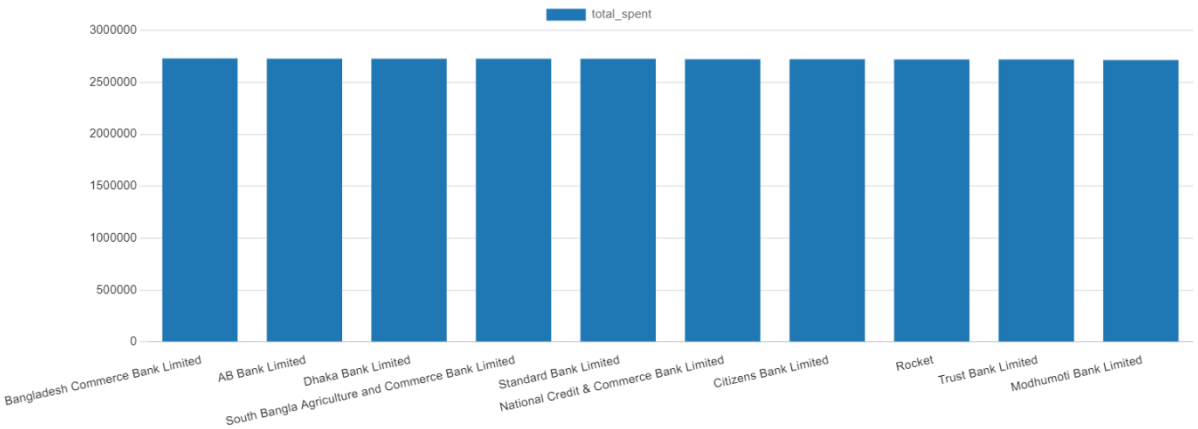
	trans_type character varying (50)	total_value numeric
1	card	94583038.50
2	mobile	8109881.50
3	cash	2708515.75



-- Top 10 Banks by Total Spent--

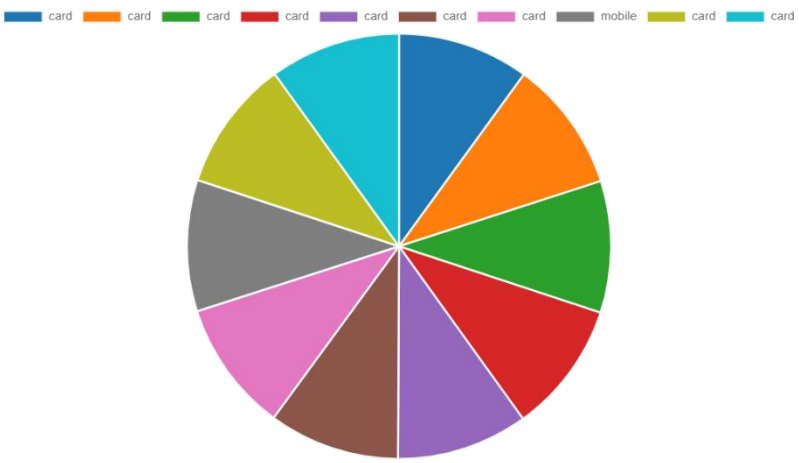
```
WITH cte AS (
    SELECT
        customer_key,
        payment_key,
        quantity,
        unit_price,
        quantity * unit_price AS total_value
    FROM fact_table
)
SELECT
    td.bank_name,
    SUM(ct.total_value) AS total_spent
FROM cte ct
JOIN trans_dim td ON ct.payment_key = td.payment_key
WHERE td.bank_name IS NOT NULL
GROUP BY td.bank_name
ORDER BY total_spent DESC
LIMIT 10;
```

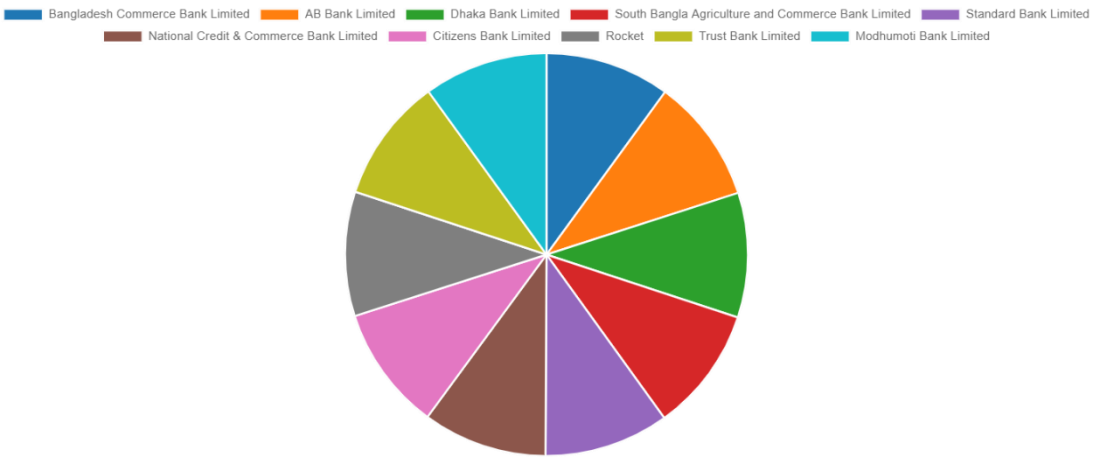
	bank_name character varying (100)	total_spent numeric
1	Bangladesh Commerce Bank Limited	2730999.25
2	AB Bank Limited	2728515.50
3	Dhaka Bank Limited	2728366.00
4	South Bangla Agriculture and Commerce Bank Limited	2728115.25



-- Top 10 Bank + Payment Method Combinations by Spending--

```
WITH cte AS (
   SELECT
      customer_key,
      payment_key,
      quantity,
      unit_price,
      quantity * unit_price AS total_value
   FROM fact_table
)
SELECT
   td.bank_name,
   td.trans_type AS payment_method,
   SUM(ct.total_value) AS total_spent
FROM cte ct
JOIN trans_dim td ON ct.payment_key = td.payment_key
WHERE td.bank_name IS NOT NULL
GROUP BY td.bank_name, td.trans_type
ORDER BY total_spent DESC
LIMIT 10;
```



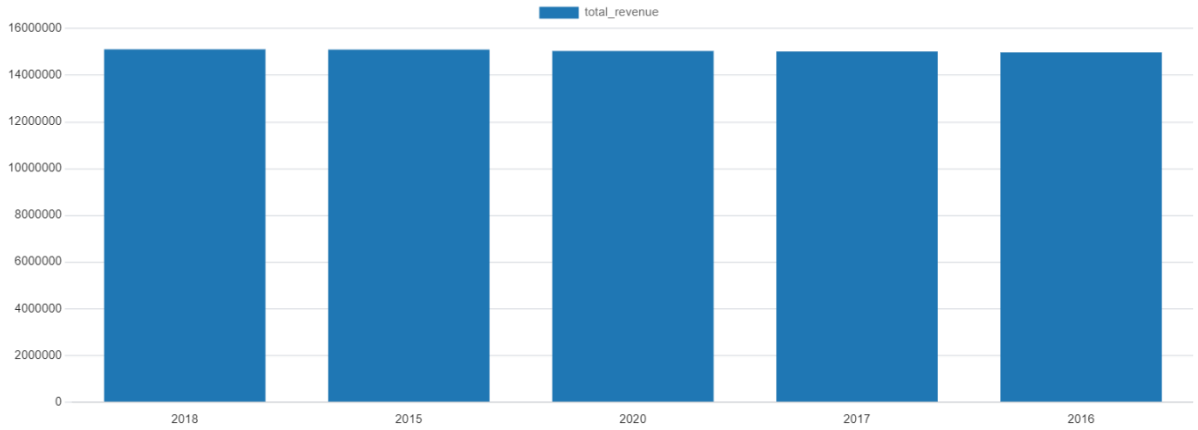


	bank_name character varying (100)	payment_method character varying (50)	total_spent numeric
1	Bangladesh Commerce Bank Limited	card	2730999.25
2	AB Bank Limited	card	2728515.50
3	Dhaka Bank Limited	card	2728366.00
4	South Bangla Agriculture and Commerce Bank Limited	card	2728115.25

--Top 5 Years by Revenue--

```
SELECT
    t.year,
    SUM(f.quantity * f.unit_price) AS total_revenue
FROM fact_table f
JOIN time_dim t ON f.time_key = t.time_key
GROUP BY t.year
ORDER BY total_revenue DESC
LIMIT 5;
```

	year integer	total_revenue numeric
1	2018	15108197.25
2	2015	15095720.25
3	2020	15037190.25


total_revenue
16000000
14000000
12000000
10000000
8000000
6000000
4000000
2000000
0
2018
2015
2020
2017
2016