

SALES INSIGHTS

1. Show all customer records:

The screenshot shows a database management interface with a tabbed view at the top containing 'customers', 'transactions', 'products', 'markets', and 'date'. The 'customers' tab is active. Below the tabs is a toolbar with various icons for file operations, editing, and viewing. A text area contains the SQL query: `1 • SELECT * FROM sales.customers;`. Below the query area is a horizontal separator line. Underneath the separator is another toolbar with options like 'Result Grid', 'Filter Rows', 'Edit', 'Export/Import', and 'Wrap Cell Content'. The main area displays a table with the following data:

	customer_code	customer_name	customer_type
▶	Cus001	Surge Stores	Brick & Mortar
	Cus002	Nomad Stores	Brick & Mortar
	Cus003	Excel Stores	Brick & Mortar
	Cus004	Surface Stores	Brick & Mortar
	Cus005	Premium Stores	Brick & Mortar
	Cus006	Electricalsara Stores	Brick & Mortar
	Cus007	Info Stores	Brick & Mortar
	Cus008	Acclaimed Stores	Brick & Mortar
	Cus009	Electricalsquipo Stores	Brick & Mortar
	Cus010	Atlas Stores	Brick & Mortar
	Cus011	Flawless Stores	Brick & Mortar
	Cus012	Integration Stores	Brick & Mortar
	Cus013	Unity Stores	Brick & Mortar
	Cus014	Forward Stores	Brick & Mortar
	Cus015	Electricalsbea Stores	Brick & Mortar
	Cus016	Logic Stores	Brick & Mortar
	Cus017	Eric Stores	Brick & Mortar

At the bottom left, there is a tab labeled 'customers 1' with a close button (X).

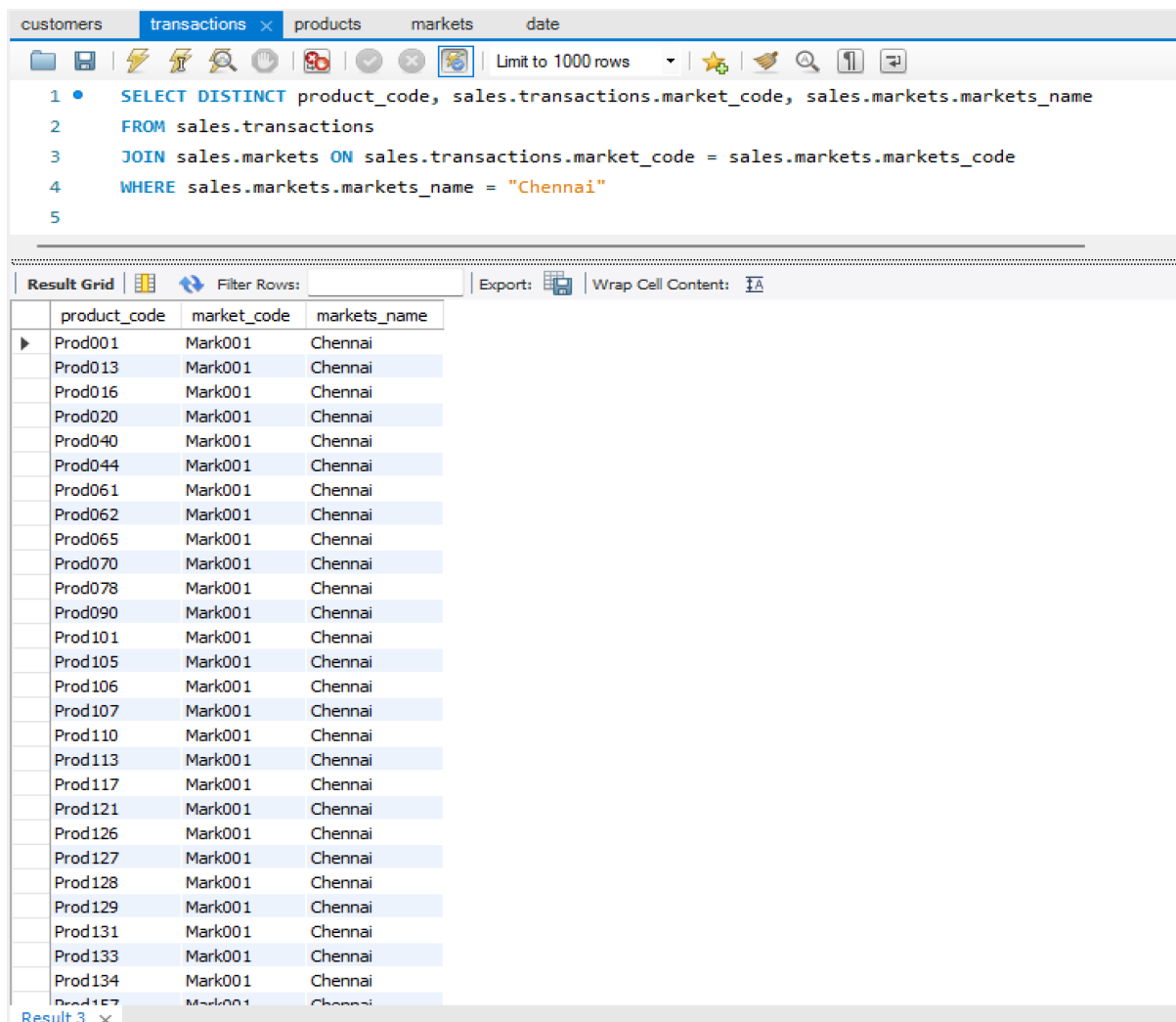
2. Show total number of customers:

customers	transactions	products	markets	date
Limit to 1000 rows				
1 SELECT COUNT(*) AS Total_customers FROM sales.customers				
Result Grid				
Filter Rows:				
Export:				
Wrap Cell Content:				
Total_customers				
38				

3. Show transactions from Chennai Market:

customers transactions x products markets date

4. Show distinct product codes that were sold in Chennai:



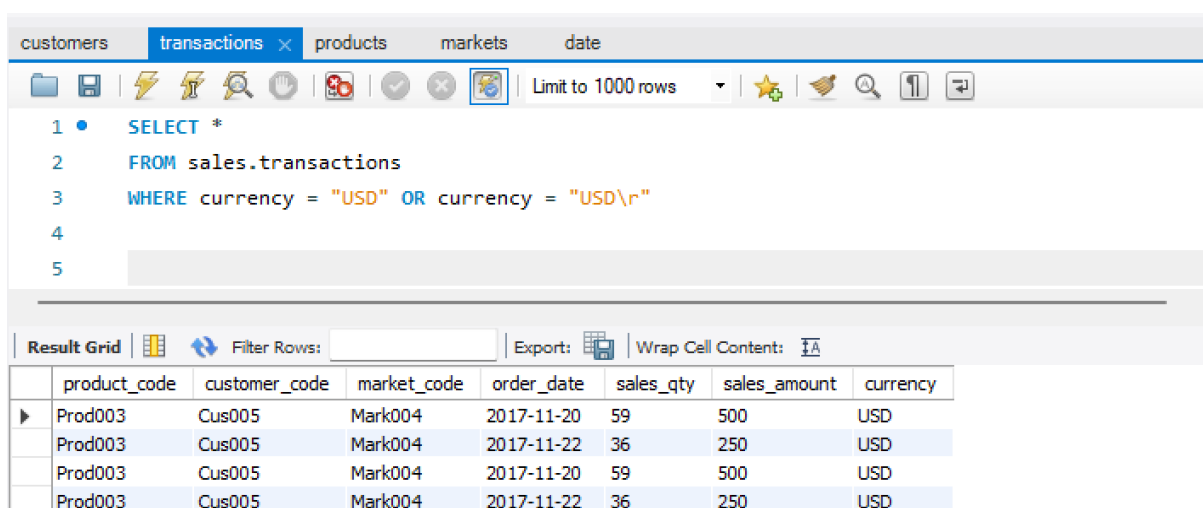
The screenshot shows a database query interface with tabs for customers, transactions, products, markets, and date. The 'transactions' tab is active. The SQL query is as follows:

```
1 • SELECT DISTINCT product_code, sales.transactions.market_code, sales.markets.markets_name
2 FROM sales.transactions
3 JOIN sales.markets ON sales.transactions.market_code = sales.markets.markets_code
4 WHERE sales.markets.markets_name = "Chennai"
5
```

The results are displayed in a table with the following columns: product_code, market_code, and markets_name. The results show 25 distinct product codes sold in Chennai, all with market_code Mark001.

product_code	market_code	markets_name
Prod001	Mark001	Chennai
Prod013	Mark001	Chennai
Prod016	Mark001	Chennai
Prod020	Mark001	Chennai
Prod040	Mark001	Chennai
Prod044	Mark001	Chennai
Prod061	Mark001	Chennai
Prod062	Mark001	Chennai
Prod065	Mark001	Chennai
Prod070	Mark001	Chennai
Prod078	Mark001	Chennai
Prod090	Mark001	Chennai
Prod101	Mark001	Chennai
Prod105	Mark001	Chennai
Prod106	Mark001	Chennai
Prod107	Mark001	Chennai
Prod110	Mark001	Chennai
Prod113	Mark001	Chennai
Prod117	Mark001	Chennai
Prod121	Mark001	Chennai
Prod126	Mark001	Chennai
Prod127	Mark001	Chennai
Prod128	Mark001	Chennai
Prod129	Mark001	Chennai
Prod131	Mark001	Chennai
Prod133	Mark001	Chennai
Prod134	Mark001	Chennai
Prod157	Mark001	Chennai

5. Show transactions where currency is US dollar:



The screenshot shows a database query interface with tabs for customers, transactions, products, markets, and date. The 'transactions' tab is active. The SQL query is as follows:

```
1 • SELECT *
2 FROM sales.transactions
3 WHERE currency = "USD" OR currency = "USD\r"
4
5
```

The results are displayed in a table with the following columns: product_code, customer_code, market_code, order_date, sales_qty, sales_amount, and currency. The results show 4 transactions with currency USD.

product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency
Prod003	Cus005	Mark004	2017-11-20	59	500	USD
Prod003	Cus005	Mark004	2017-11-22	36	250	USD
Prod003	Cus005	Mark004	2017-11-20	59	500	USD
Prod003	Cus005	Mark004	2017-11-22	36	250	USD

6. Show transactions in 2020 join by date table:

customers transactions products markets date

Limit to 1000 rows

```
1 • SELECT sales.transactions.*, sales.date.year
2 FROM sales.transactions
3 JOIN sales.date ON sales.transactions.order_date = sales.date.date
4 WHERE sales.date.year = 2020
5
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

	product_code	customer_code	market_code	order_date	sales_qty	sales_amount	currency	year
▶	Prod005	Cus007	Mark004	2020-01-09	1	630	INR	2020
	Prod005	Cus007	Mark004	2020-01-10	1	472	INR	2020
	Prod005	Cus007	Mark004	2020-01-17	2	2042	INR	2020
	Prod005	Cus007	Mark004	2020-02-07	1	417	INR	2020
	Prod005	Cus007	Mark004	2020-02-14	1	310	INR	2020
	Prod005	Cus007	Mark004	2020-02-28	1	208	INR	2020
	Prod005	Cus007	Mark004	2020-03-06	1	620	INR	2020
	Prod005	Cus007	Mark004	2020-03-13	1	620	INR	2020
	Prod005	Cus007	Mark004	2020-04-03	1	829	INR	2020
	Prod005	Cus007	Mark004	2020-04-14	4	2694	INR	2020
	Prod005	Cus007	Mark004	2020-04-20	1	102	INR	2020
	Prod005	Cus007	Mark004	2020-05-15	2	528	INR	2020
	Prod011	Cus016	Mark002	2020-06-12	1	1028	INR	2020
	Prod011	Cus016	Mark002	2020-06-16	1	514	INR	2020
	Prod005	Cus007	Mark004	2020-01-09	1	630	INR	2020
	Prod005	Cus007	Mark004	2020-01-10	1	472	INR	2020
	Prod005	Cus007	Mark004	2020-01-17	2	2042	INR	2020
	Prod005	Cus007	Mark004	2020-02-07	1	417	INR	2020
	Prod005	Cus007	Mark004	2020-02-14	1	310	INR	2020
	Prod005	Cus007	Mark004	2020-02-28	1	208	INR	2020
	Prod005	Cus007	Mark004	2020-03-06	1	620	INR	2020
	Prod005	Cus007	Mark004	2020-03-13	1	620	INR	2020
	Prod005	Cus007	Mark004	2020-04-03	1	829	INR	2020
	Prod005	Cus007	Mark004	2020-04-14	4	2694	INR	2020
	Prod005	Cus007	Mark004	2020-04-20	1	102	INR	2020
	Prod005	Cus007	Mark004	2020-05-15	2	528	INR	2020
	Prod011	Cus016	Mark002	2020-06-12	1	1028	INR	2020
	Prod011	Cus016	Mark002	2020-06-16	1	514	INR	2020

Result 11 x

7. Show total revenue in each year:

customers transactions products markets date

Limit to 1000 rows

```
1 • SELECT SUM(sales.transactions.sales_amount) AS Total_Revenue, sales.date.year
2 FROM sales.transactions
3 JOIN sales.date ON sales.transactions.order_date = sales.date.date
4 GROUP BY sales.date.year
5 ORDER BY Total_Revenue DESC
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: |

	Total_Revenue	year
▶	414308941	2018
	336452114	2019
	142235559	2020
	93569152	2017

8. Show month-wise total revenue in the year 2020:

The screenshot shows a SQL IDE interface with tabs for customers, transactions, products, markets, and date. The 'transactions' tab is active. The query editor contains the following SQL code:

```
1 • SELECT SUM(sales.transactions.sales_amount) AS Total_Revenue, sales.date.year, sales.date.month_name
2 FROM sales.transactions
3 JOIN sales.date ON sales.transactions.order_date = sales.date.date
4 WHERE sales.date.year = 2020
5 GROUP BY sales.date.month_name
6 ORDER BY Total_Revenue DESC
```

Below the query editor, the 'Result Grid' shows the results of the query. The columns are Total_Revenue, year, and month_name. The results are sorted by Total_Revenue in descending order.

Total_Revenue	year	month_name
26925734	2020	February
26385217	2020	March
25659711	2020	January
25264333	2020	April
23288588	2020	May
14711976	2020	June

9. Show total revenue in each city in the year 2020:

The screenshot shows a SQL IDE interface with tabs for customers, transactions, products, markets, and date. The 'transactions' tab is active. The query editor contains the following SQL code:

```
1 • SELECT SUM(sales.transactions.sales_amount) AS Total_Revenue, sales.date.year, sales.markets.markets_name
2 FROM sales.transactions
3 JOIN sales.date ON sales.transactions.order_date = sales.date.date
4 JOIN sales.markets ON sales.transactions.market_code = sales.markets.markets_code
5 WHERE sales.date.year = 2020
6 GROUP BY sales.markets.markets_name
7 ORDER BY Total_Revenue DESC
```

Below the query editor, the 'Result Grid' shows the results of the query. The columns are Total_Revenue, year, and markets_name. The results are sorted by Total_Revenue in descending order.

Total_Revenue	year	markets_name
77742074	2020	Delhi NCR
20183077	2020	Mumbai
18011939	2020	Ahmedabad
8254690	2020	Nagpur
7662003	2020	Bhopal
2727273	2020	Kochi
2463024	2020	Chennai
2179096	2020	Kanpur
1208652	2020	Hyderabad
895777	2020	Patna
495760	2020	Lucknow
250623	2020	Surat
161571	2020	Bhubaneshwar

10. Show total Revenue by customers:











customers

transactions






products

markets

date





Limit to 1000 rows




```
1 • SELECT SUM(sales.transactions.sales_amount) AS Total_Revenue, sales.customers.custmer_name
2 FROM sales.transactions
3 JOIN sales.customers on sales.transactions.customer_code = sales.customers.customer_code
4 GROUP BY sales.customers.custmer_name
5 ORDER BY Total_Revenue DESC
6
7
```

Result Grid




Filter Rows:

Export:



Wrap Cell Content:



	Total_Revenue	custmer_name
▶	413905769	Electricalsara Stores
	49644189	Electricalslytical
	49175285	Excel Stores
	45258250	Premium Stores
	43916981	Nixon
	35359233	Info Stores
	31771997	Control
	28833717	Surge Stores
	21198041	Acclaimed Stores
	21079123	Forward Stores
	18794634	Epic Stores
	17739349	Nomad Stores
	17489935	Electricalsocity
	17379851	Modular
	16716803	Atlas Stores
	16529970	Leader
	15249738	Surface Stores
	13993708	Integration Stores
	13264523	Logic Stores
	12995938	Path
	12618892	Unity Stores
	10310851	Electricalsopedia St...
	9162106	Flawless Stores
	6173068	Synthetic
	6068432	All-Out

Result 19 ×