# Project Dillard's – Managing Big Data with TERADATA

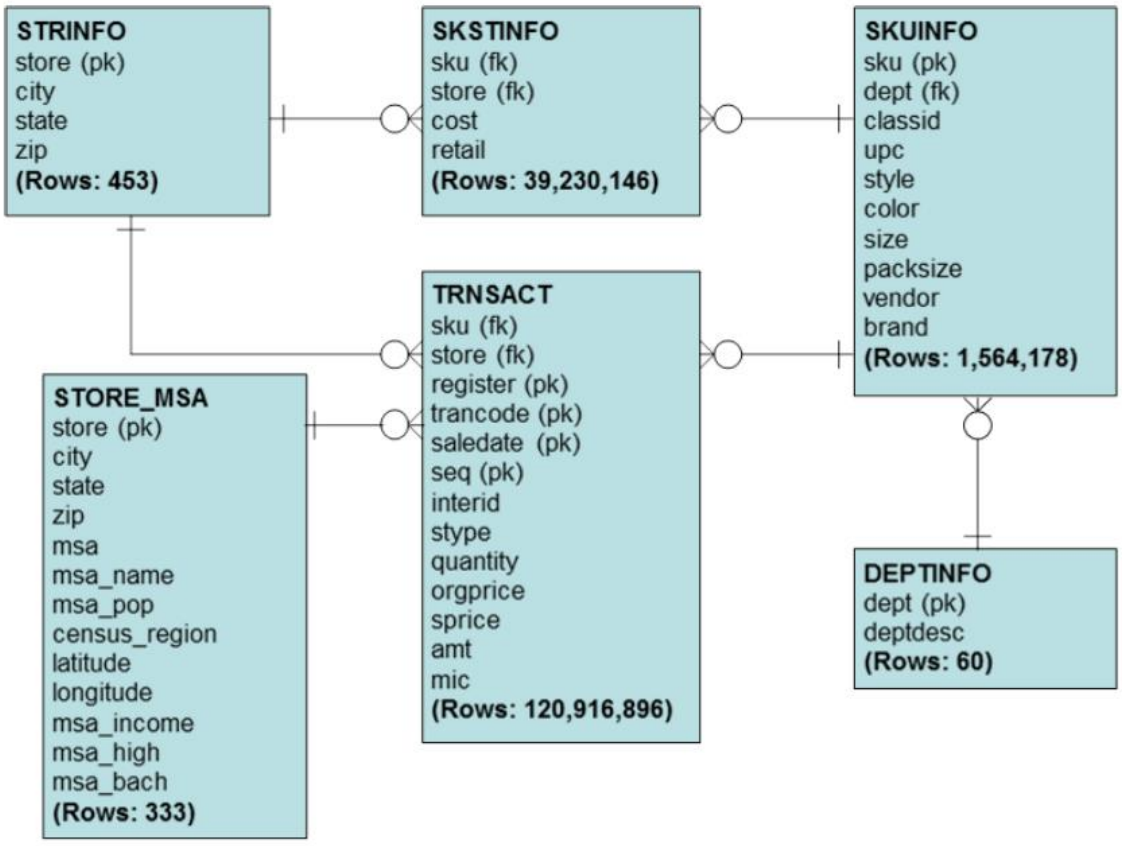
## Introduction

The project was part of Coursera Course: Managing Big Data with MySQL / TERADATA, offered by Duke University. Dillard's donated this data set so that students in the course can have the opportunity to practice querying and analyzing a real enterprise data set. The Dillard's dataset is stored in the Teradata database and students were able to write queries in Teradata Viewpoint Scratchpad.

The data set contains a year of Dillard's sales transaction data from August 2004 through July 2005. The information is spread across six tables ranging from 60 to over 120 million rows and size. The transaction table includes what items were included in each transaction, how much they cost, when the purchase was made, and where the purchase was made. Additionally, there are census information about the metropolitan's statistical area surrounding the store in which the purchase was made.

## Relational Schema

The following diagram represents the relational schema of the date set.



## Basic Descriptions of the data set

- Transaction Table

It contains information such as store numbers, sku numbers, types of transaction, sale dates, original prices, sale prices, quantities, amount, etc.

There are two types of transactions: purchases and returns. We will need to make sure we specify in which type we are interested, when running queries using the transaction table.

- Skuinfo Table

It contains information such as brand, color, department, etc.

- Skstinfo Table

It contains information such as skus, store, cost and retail price.

- Deptinfo Table

It contains department numbers, and department names.

- Strinfo Table

It contains information store numbers, the cities and states they were located.

- Store_msa Table

It contains population statistics about the geographic location around a store.

There are strange entries that likely reflect entry errors, such as rows that have “0” in their orgprice column, rows in the skstinfo table where both the cost and retail price are listed as 0.00, rows in the skstinfo table where the cost is greater than the retail price.

There are a lot of strange values in the “color”, “style”, and “size” fields of the skuinfo table. For example there are entries like "BMK/TOUR K” and “ALOE COMBO” in the color field, even though those entries do not represent colors.

## Questions we are interested in

In this project, we want to identify whether the characteristics of the geographic location in which a store resides correlates with the sales performance of the store. If it does, we can design strategies that will allow Dillard’s to take advantage of the geographic trends, or make decisions about how to handle geographic locations that consistently have poor sales performance.

In addition, we want to identify which months had a better sales performance and whether this sales trend is consistent so that we can make recommendations to Dillard’s to design their marketing and inventory strategies appropriately.

# Queries and Analysis

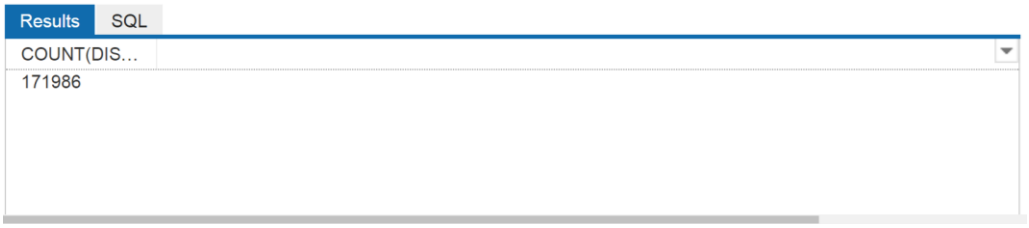
## Part One

The first part examines Dillard's sales primarily by adding up total amounts of revenue, and by calculating the profit to get a sense of general sales trends.

Comparing distinct skus in the skuinfo, skstinfo, and trnsact tables, we notice that # distinct skus in skuinfo > # distinct skus in skstinfo > # distinct skus in trnsact

```
SELECT COUNT(DISTINCT sku)
FROM skuinfo;
SELECT COUNT(DISTINCT sku)
FROM skstinfo;
SELECT COUNT(DISTINCT sku)
FROM trnsact;

SELECT COUNT(DISTINCT t.sku)
FROM trnsact t  LEFT JOIN skstinfo s ON t.sku=s.sku
WHERE t.sku IS NOT NULL AND s.sku IS NULL;
```




It turns out that there are many skus in the trnsact table that are not skstinfo table. Since we do not have the cost information for all the skus in the transact table, we will not be able to complete many desirable analyses of Dillard's profit, as opposed to revenue.

Although we can't complete all the analyses on Dillard's profit, we can look at general trends.

- The average amount of profit Dillard's made per day is $1,527,903

```
SELECT SUM(t.amt - t.quantity*s.cost)/COUNT(DISTINCT t.saledate) AS AVGProfit
FROM trnsact t LEFT JOIN SKSTINFO s
ON t.sku = s.sku AND t.store = s.store
WHERE t.stype = 'p';
```

- Dillard's income based on total sum of purchases the greatest on December 18, 2004.

```
SELECT TOP 5 saledate, SUM(amt) AS tot_pur
FROM trnsact
WHERE stype='p'
Group by saledate
Order by tot_pur DESC;
```

Results | SQL

| SALEDATE | TOT_PUR |
|---|---|
| 04/12/18 | 19813655.17 |
| 04/12/23 | 18922032.61 |
| 05/02/26 | 17622004.35 |
| 04/11/26 | 17425341.39 |
| 04/12/21 | 16479604.61 |

- Clinique's Dramatically Different Moisturizing Lotion brought in the most revenue.

```
SELECT TOP 10 b.dept, c.deptdesc, b.brand, b.style, b.color, SUM(a.amt) AS tot_sales
FROM trnsact a, skuinfo b, deptinfo c
WHERE a.sku=b.sku AND b.dept=c.dept AND a.stype='p'
GROUP BY b.dept, c.deptdesc, b.brand, b.style, b.color
ORDER BY tot_sales DESC;
```

Results | SQL

| DEPT | DEPTDESC | BRAND | STYLE | COLOR | TOT_SALES |
|---|---|---|---|---|---|
| 800 | CLINIQUE | CLINIQUE | 6142 | DDML | 6350866.72 |
| 800 | CLINIQUE | CLINIQUE | 68LE | DDML PUMP | 5828939.04 |
| 2200 | CELEBRT | LANCOME | 2410 | 01-BLACK | 4992617.69 |
| 800 | CLINIQUE | CLINIQUE | 6121 | CLARIFY | 3599194.54 |
| 2200 | CELEBRT | LANCOME | 4408 | 01-BLACK | 3444989.07 |
| 800 | CLINIQUE | CLINIQUE | 61CE01 | HAPPY 3.4OZ | 3049120.83 |
| 800 | CLINIQUE | CLINIQUE | 645J | 02NEUTRAL | 2749931.03 |
| 2200 | CELEBRT | LANCOME | 2101 | ABSOLUE CRM | 2692700.04 |
| 800 | CLINIQUE | CLINIQUE | 67GF | TOTAL TURNAR | 2555136.78 |
| 6400 | BLUE | DESIGNER | 6 7002-9 | EDP SPRAY | 2410574.64 |

10 rows total

- The store that had the greatest total revenue located at Metairie, LA.

```
SELECT TOP 5 a.store, b.city, b.state, SUM(a.amt) AS tot_sales
FROM trnsact a JOIN strinfo b
ON a.store=b.store
WHERE a.stype='P'
GROUP BY a.store, b.state, b.city
ORDER BY tot_sales DESC;
```

Results SQL

| STORE | CITY | STATE | TOT_SALES |
|---|---|---|---|
| 8402 | METAIRIE | LA | 24171426.58 |
| 504 | LITTLE ROCK | AR | 22792579.65 |
| 2707 | MCALLEN | TX | 22331884.55 |
| 1607 | DALLAS | TX | 22063797.73 |
| 9103 | LOUISVILLE | KY | 20114154.20 |

## Part Two

Next, we looked at sales trends across stores and months to determine if there are specific stores, departments, or times of year that are associated with better or worse sales performance.

- The first thing we noticed is that there are 27 days recorded in the database during August, 2005, but 31 days recorded in the database during August, 2004 when examining month/year combinations in the database.
  Since August is the only month that is repeated in our dataset and August, 2005 data is curtailed, we will restrict our analysis of August sales to those recorded in 2004.

```
SELECT EXTRACT(YEAR from saledate) AS sales_year,
EXTRACT(MONTH from saledate) AS sales_month,
COUNT(DISTINCT saledate) as numdays
FROM trnsact
GROUP BY sales_year, sales_month
ORDER BY sales_year, sales_month;
```

Results SQL

| SALES_YEAR | SALES_MO... | NUMDAYS |
|---|---|---|
| 2004 | 8 | 31 |
| 2004 | 9 | 30 |
| 2004 | 10 | 31 |
| 2004 | 11 | 29 |
| 2004 | 12 | 30 |
| 2005 | 1 | 31 |
| 2005 | 2 | 28 |
| 2005 | 3 | 30 |
| 2005 | 4 | 30 |
| 2005 | 5 | 31 |
| 2005 | 6 | 30 |
| 2005 | 7 | 31 |
| 2005 | 8 | 27 |

13 rows total

- We found there are many month/year/store combinations that only have one day of transaction data stored in the database. The possible reason is that Dillard's may have removed some data before donating it. We will need to take these missing data into account in many of our future analyses, especially analyses that aim to compare sales trends within subsets of stores.

```
SELECT EXTRACT(YEAR from saledate) AS sales_year,
EXTRACT(MONTH from saledate) AS sales_month,
store, COUNT(DISTINCT saledate) as numdays
FROM trnsact
GROUP BY sales_year, sales_month, store
ORDER BY numdays;
```

Results | SQL

| SALES_YEAR | SALES_MO... | STORE | NUMDAYS |
|---|---|---|---|
| 2005 | 7 | 7604 | 1 |
| 2004 | 8 | 9906 | 1 |
| 2004 | 8 | 8304 | 1 |
| 2004 | 9 | 4402 | 1 |
| 2005 | 3 | 8304 | 1 |
| 2004 | 8 | 7203 | 3 |
| 2005 | 3 | 6402 | 11 |
| 2005 | 4 | 5703 | 16 |
| 2005 | 3 | 3002 | 16 |
| 2004 | 12 | 1804 | 17 |
| 2004 | 10 | 4903 | 17 |
| 2004 | 11 | 4402 | 17 |
| 2004 | 9 | 3802 | 21 |
| 2005 | 4 | 1704 | 21 |

Page 1 of 85 (4225 rows total)

- Since there are different numbers of days in each month of the year, we will takes the number of days into account and assess sales trends by summing the total revenue for a given time period, and dividing by the total number of days that contributed to that time period, as opposed to simply adding up all the sales in each month.
  Because we only want to include data from store/month/year combinations that have enough data to justify taking an average, we only examine store/month/year combinations that have at least 20 days of data within that month.

```
SELECT EXTRACT(YEAR from saledate)||EXTRACT(MONTH from saledate),store,
SUM(amt)/COUNT(DISTINCT saledate) AS daily_avg
FROM trnsact
WHERE stype='P' AND (EXTRACT(YEAR from saledate) <> 2005 OR  EXTRACT(MONTH
from saledate) <> 8)
GROUP BY EXTRACT(YEAR from saledate)||EXTRACT(MONTH from saledate),store
ORDER BY EXTRACT(YEAR from saledate)||EXTRACT(MONTH from saledate),store
HAVING COUNT(DISTINCT saledate) >=20;
```

Results SQL

| (EXTRACT(... | STORE | DAILY_AVG |
|---|---|---|
| 2004 8 | 102 | 29385.04 |
| 2004 8 | 103 | 23817.66 |
| 2004 8 | 107 | 37698.96 |
| 2004 8 | 202 | 14347.24 |
| 2004 8 | 203 | 22087.60 |
| 2004 8 | 204 | 12369.21 |
| 2004 8 | 209 | 27000.87 |
| 2004 8 | 302 | 20103.28 |
| 2004 8 | 303 | 8705.70 |
| 2004 8 | 304 | 27210.64 |
| 2004 8 | 307 | 14891.89 |
| 2004 8 | 309 | 12056.69 |
| 2004 8 | 402 | 24774.56 |
| 2004 8 | 403 | 6131.50 |

Page 1 of 78 (3888 rows total)

- What was the average daily revenue Dillard's brought in during each month of the year?

```
SELECT store_rev.s_year, store_rev.s_month,
SUM(store_rev.tot_sales)/SUM(store_rev.numdays) AS daily_avg
FROM
(SELECT EXTRACT(YEAR from t.saledate) AS s_year, EXTRACT(MONTH from t.saledate)
AS s_month, COUNT(DISTINCT t.saledate) AS numdays, t.store, SUM(t.amt) AS tot_sales,
CASE WHEN EXTRACT(YEAR from t.saledate)=2005 AND EXTRACT(MONTH from
t.saledate)=8 THEN 'exclude' END AS exclude_flag
FROM trnsact t
WHERE t.stype='p' AND exclude_flag IS NULL
GROUP BY s_year, s_month, t.store
HAVING numdays>=20
) AS store_rev
GROUP BY store_rev.s_year, store_rev.s_month
ORDER BY daily_avg;
```

Results SQL

| S_YEAR | S_MONTH | DAILY_AVG |
|---|---|---|
| 2004 | 8 | 17384.39 |
| 2004 | 9 | 17611.98 |
| 2005 | 1 | 18022.09 |
| 2004 | 10 | 18893.60 |
| 2004 | 11 | 19404.96 |
| 2005 | 6 | 20140.48 |
| 2005 | 5 | 20585.24 |
| 2005 | 3 | 20633.23 |
| 2005 | 4 | 21401.67 |
| 2005 | 7 | 22473.38 |
| 2005 | 2 | 22745.18 |
| 2004 | 12 | 34981.66 |

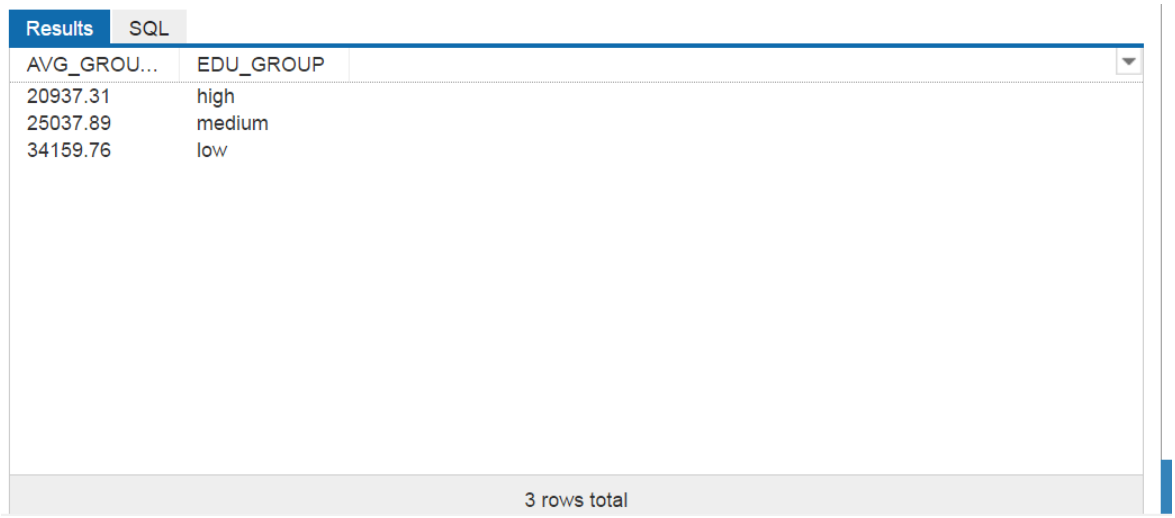
12 rows total

## Analyzing population statistics effects

The next set of questions we will explore focus on how the population statistics of the geographical location surrounding a store relate to sales performance.

- What is the average daily revenue brought in by Dillard's stores in areas of high, medium, or low levels of high school education?
  We define areas of "low" education as those that have high school graduation rates between 50-60%, areas of "medium" education as those that have high school graduation rates between 60-70%, and areas of "high" education as those that have high school graduation rates of above 70%.
  The results show that the area of high level of high school education has the lowest average daily revenue whereas the area with low level of high school education has the highest average daily revenue.

```
SELECT SUM(revenue_per_store.revenue)/SUM(numdays) AS avg_group_revenue,
CASE WHEN revenue_per_store.msa_high BETWEEN 50 AND 60 THEN 'low'
WHEN revenue_per_store.msa_high BETWEEn 60 AND 70 THEN 'medium'
WHEN revenue_per_store.msa_high>70 THEN 'high'
END as edu_group
FROM
(SELECT m.msa_high, t.store,
CASE WHEN EXTRACT(YEAR from t.saledate)=2005 AND EXTRACT(MONTH from
t.saledate)=8 THEN 'exclude' END AS exclude_flag,
SUM(t.amt) AS revenue, COUNT(DISTINCT t.saledate) AS numdays,
EXTRACT(MONTH from t.saledate) as sales_month
FROM store_msa m JOIN trnsact t
ON m.store=t.store
WHERE t.stype='p' AND exclude_flag IS NULL AND
t.store||EXTRACT(YEAR from t.saledate)||EXTRACT(MONTH from t.saledate) IN
(
SELECT store||EXTRACT(YEAR from saledate)||EXTRACT(MONTH from saledate)
FROM trnsact
GROUP BY store, EXTRACT(YEAR from saledate),EXTRACT(MONTH from saledate)
HAVING COUNT(DISTINCT saledate)>=20
)
GROUP BY t.store, m.msa_high, sales_month, exclude_flag
) AS revenue_per_store
GROUP BY edu_group
ORDER BY avg_group_revenue;
```



Results | SQL

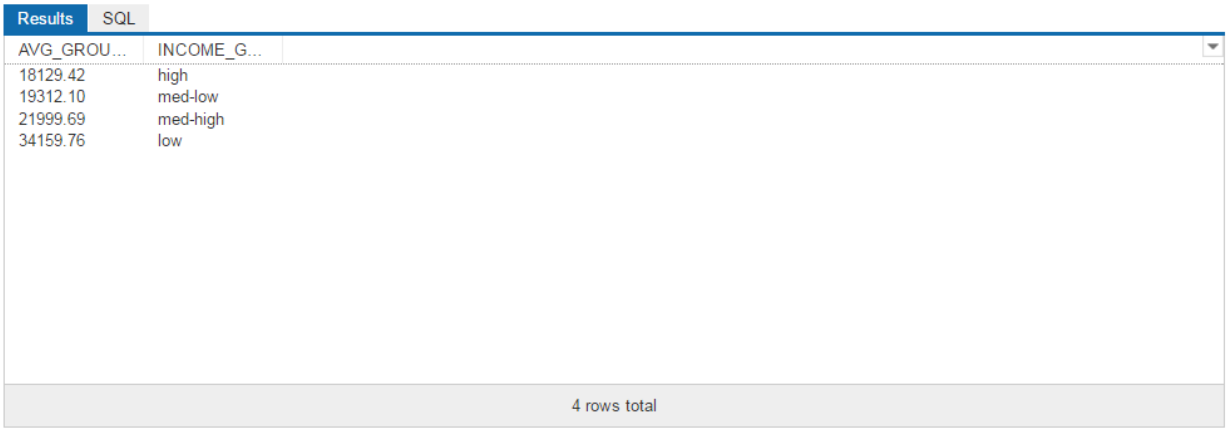
| AVG_GROU... | EDU_GROUP |
|---|---|
| 20937.31 | high |
| 25037.89 | medium |
| 34159.76 | low |

3 rows total

- Similarly, we also found low msa_income group has the highest average daily revenue per store.
  We divided the msa_income groups so that msa_incomes between 1 and 20,000 are labeled 'low', msa_incomes between 20,001 and 30,000 are labeled 'med-low', msa_incomes between 30,001 and 40,000 are labeled 'med-high',and msa_incomes between 40,001 and 60,000 are labeled 'high'.

```
SELECT SUM(revenue_per_store.revenue)/SUM(numdays) AS avg_group_revenue,
CASE WHEN revenue_per_store.msa_income BETWEEN 1 AND 20000 THEN 'low'
WHEN revenue_per_store.msa_income BETWEEn 20001 AND 30000 THEN 'med-low'
WHEN revenue_per_store.msa_income BETWEEn 30001 AND 40000 THEN 'med-high'
WHEN revenue_per_store.msa_income BETWEEn 40001 AND 60000 THEN 'high'
END as income_group
FROM
(SELECT m.msa_income, t.store,
CASE WHEN EXTRACT(YEAR from t.saledate)=2005 AND EXTRACT(MONTH from
t.saledate)=8 THEN 'exclude' END AS exclude_flag,
SUM(t.amt) AS revenue, COUNT(DISTINCT t.saledate) AS numdays,
EXTRACT(MONTH from t.saledate) as sales_month
FROM store_msa m JOIN trnsact t
ON m.store=t.store
WHERE t.stype='p' AND exclude_flag IS NULL AND
t.store||EXTRACT(YEAR from t.saledate)||EXTRACT(MONTH from t.saledate) IN
(
SELECT store||EXTRACT(YEAR from saledate)||EXTRACT(MONTH from saledate)
FROM trnsact
GROUP BY store, EXTRACT(YEAR from saledate),EXTRACT(MONTH from saledate)
HAVING COUNT(DISTINCT saledate)>=20
)
GROUP BY t.store, m.msa_income, sales_month, exclude_flag
```

```
) AS revenue_per_store
GROUP BY income_group
ORDER BY avg_group_revenue;
```



Results | SQL

| AVG_GROU... | INCOME_G... |
|---|---|
| 18129.42 | high |
| 19312.10 | med-low |
| 21999.69 | med-high |
| 34159.76 | low |

4 rows total

- Next, we compared the average daily revenues of the stores with the highest median msa_income and the lowest median msa_income.
  The store with the highest median msa_income was in Spanish Fort, AL. It had a lower average daily revenue than the store with the lowest median msa_income, which was in McAllen, TX.

```
SELECT SUM(store_rev.tot_sales)/SUM(store_rev.numdays) AS daily_avg,
store_rev.msa_income as med_income, store_rev.city, store_rev.state
FROM
(SELECT COUNT(DISTINCT t.saledate) AS numdays,
EXTRACT(YEAR from t.saledate) as sales_year,
EXTRACT(MONTH from t.saledate) as sales_month,
t.store, SUM(t.amt) AS tot_sales,
CASE WHEN EXTRACT(YEAR from t.saledate)=2005 AND EXTRACT(MONTH from
t.saledate)=8 THEN 'exclude' END AS exclude_flag, m.msa_income, s.city, s.state
FROM trnsact t JOIN store_msa m
ON m.store=t.store JOIN strinfo s
ON t.store=s.store
WHERE t.stype='P' AND exclude_flag IS NULL
GROUP BY sales_year, sales_month, t.store, m.msa_income, s.city, s.state
HAVING numdays>=20 ) AS store_rev
WHERE store_rev.msa_income IN
((SELECT MAX(msa_income) FROM store_msa),
(SELECT MIN(msa_income) FROM store_msa))
GROUP BY med_income, store_rev.city, store_rev.state;
```

Results SQL

| DAILY_AVG | MED_INCO... | CITY | STATE |
|---|---|---|---|
| 17884.08 | 56099 | SPANISH FORT | AL |
| 56601.99 | 16022 | MCALLEN | TX |

2 rows total

- Stores in a larger population msa seems to have larger average daily revenue.

  We divided stores up so that stores with msa population between 1 and 100,000 are labeled 'very small', store with msa population between 100,001 and 200,000 are labeled 'small', store with msa population between 200,001 and 500,000 are labeled 'med-small', store with msa population between 500,001 and 1,000,000 are labeled 'med-large', store with msa population between 1,000,001 and 5,000,000 are labeled 'very large'.

```
SELECT SUM(store_rev.tot_sales)/SUM(store_rev.numdays) AS daily_avg,
CASE WHEN store_rev.msa_pop BETWEEN 1 AND 100000 THEN 'very small'
WHEN store_rev.msa_pop BETWEEN 100001 AND 200000 THEN 'small'
WHEN store_rev.msa_pop BETWEEN 200001 AND 500000 THEN 'med-small'
WHEN store_rev.msa_pop BETWEEN 500001 AND 1000000 THEN 'med-large'
WHEN store_rev.msa_pop BETWEEN 1000001 AND 5000000 THEN 'large'
WHEN store_rev.msa_pop>5000000 THEN 'very large'
END as pop_group
FROM
(SELECT m.msa_pop, EXTRACT(YEAR from t.saledate) AS s_year, EXTRACT(MONTH from
t.saledate) AS s_month, COUNT(DISTINCT t.saledate) AS numdays, t.store, SUM(t.amt)
AS tot_sales,
CASE WHEN EXTRACT(YEAR from t.saledate)=2005 AND EXTRACT(MONTH from
t.saledate)=8 THEN 'exclude' END AS exclude_flag
FROM store_msa m JOIN trnsact t ON m.store=t.store
WHERE t.stype='p' AND exclude_flag IS NULL
GROUP BY s_year, s_month, t.store, m.msa_pop
HAVING numdays>=20
) AS store_rev
```

```
GROUP BY pop_group
ORDER BY daily_avg;
```

Results SQL

| DAILY_AVG | POP_GROUP |
|---|---|
| 12688.25 | very small |
| 16355.16 | small |
| 21208.43 | med-small |
| 22107.57 | large |
| 24341.59 | med-large |
| 25451.53 | very large |

6 rows total

## Analyzing monthly (or seasonal) sales effects

- Now we assess the sales performance of individual SKU numbers by summing the total revenue associated with each SKU. Which SKU number had the greatest increase in total sales revenue from November to December?
  We will NOT exclude data when examining the sales performance associated with individual merchandise items because different stores will sell different numbers of items at different times so we have no reason for assuming the missing data will disproportionately affect any one SKU.
  The SKU number 3949538 had the greatest increase in total sales revenue from November to December.

```
SELECT sku,
SUM(CASE WHEN EXTRACT(MONTH from saledate)=11 THEN amt END) AS November,
SUM(CASE WHEN EXTRACT(MONTH from saledate)=12 THEN amt END) AS December,
December-November AS sales_bump
FROM trnsact
WHERE stype='p'
GROUP BY sku
ORDER BY sales_bump DESC;
```

Results SQL

| SKU | NOVEMBER | DECEMBER | SALES_BU... |
|---|---|---|---|
| 3949538 | 121176.70 | 936256.91 | 815080.21 |
| 2637537 | 178632.50 | 844448.20 | 665815.70 |
| 994478 | 253186.16 | 899066.21 | 645880.05 |
| 1310252 | 100875.50 | 673971.30 | 573095.80 |
| 4737469 | 142449.53 | 662113.42 | 519663.89 |
| 944478 | 124888.50 | 617358.23 | 492469.73 |
| 6966816 | 59474.00 | 548789.50 | 489315.50 |
| 173088 | 111577.49 | 557335.75 | 445758.26 |
| 457542 | 73694.00 | 495848.63 | 422154.63 |
| 1297499 | 66414.51 | 470338.83 | 403924.32 |
| 637738 | 135184.50 | 529617.82 | 394433.32 |
| 2938210 | 1600.00 | 395994.00 | 394394.00 |
| 6926816 | 80066.10 | 440401.25 | 360335.15 |
| 9288505 | 51381.00 | 383928.25 | 332547.25 |

Page 1 of 14264 (713172 rows total)

- We calculated the standard deviation of the price of a SKU to give us an idea of the variability of the sale prices of that item.
  A high standard deviation could indicate that the item often needs to be put on sale (the price lowered from the normal selling price) for customers to buy it (which could indicate that the original price set by the manufacturer or retailer is too high), or that it has been discontinued. It could also indicate that stores in different parts of the country have priced the item very differently, either intentionally due to marketing strategies tailored for specific geographic locations, or due to error.
  High standard deviations could also simply indicate that there are many mistakes in the database for the SKU so that parts of the dataset need to be cleaned.
  We only examined SKUs which have been part of over 100 transactions.

```
SELECT DISTINCT top10skus.sku, top10skus.sprice_stdev, top10skus.num_transactions,
si.style, si.color,si.size, si.packsize,si.vendor,si.brand
FROM
(SELECT TOP 10 sku, STDDEV_POP(sprice) AS sprice_stdev,
COUNT(sprice) as num_transactions
FROM trnsact
WHERE stype='p'
GROUP BY sku
HAVING num_transactions>100
ORDER BY sprice_stdev DESC) AS top10skus JOIN skuinfo si
ON top10skus.sku=si.sku
ORDER BY top10skus.sprice_stdev DESC;
```

Results | SQL

| SKU | SPRICE_S... | NUM_TRA... | STYLE | COLOR | SIZE | PACKS... | VENDOR | BRAND |
|---|---|---|---|---|---|---|---|---|
| 2762683 | 175 | 106 | 403154133510 | BLACK | 42REG | 1 | 7045883 | HART SCH |
| 5453849 | 169.1 | 284 | 9HA 726680 | FA02 | L | 1 | 5715232 | POLO FAS |
| 5623849 | 164 | 187 | 9HA 726680 | FA02 | M | 1 | 5715232 | POLO FAS |
| 6039654 | 154.2 | 122 | 714154105423 | BLACK | 46REG | 1 | 7045883 | HART SCH |
| 4213926 | 154 | 133 | 5BRDFL726682 | FA06 | M | 1 | 5715232 | POLO FAS |
| 5719654 | 153.4 | 189 | 714154105423 | BLACK | 42REG | 1 | 7045883 | HART SCH |
| 5939654 | 153.3 | 166 | 714154105423 | BLACK | 44REG | 1 | 7045883 | HART SCH |
| 1159657 | 149.8 | 151 | 741154105423 | NAVY | 46REG | 1 | 7045883 | HART SCH |
| 5889654 | 148.7 | 123 | 714154105423 | BLACK | 44LONG | 1 | 7045883 | HART SCH |
| 9716299 | 147 | 118 | 7RLSS 736852 | TAN (X) | XL | 1 | 5715232 | POLO FAS |

10 rows total

- When examining the average daily revenue Dillard's brought in during each month of the year, we notice that December consistently has the best sales.
  The Louisvl department at Salina, KS had the greatest percent increase in average daily sales revenue from November to December.
  We only examine departments whose total sales were at least $1,000 in both November and December.

```
SELECT strinfo.store, strinfo.city, strinfo.state, d.deptdesc,
```

```
SUM(CASE WHEN EXTRACT(MONTH from saledate)=11 THEN amt END) AS November,
COUNT(DISTINCT(CASE WHEN EXTRACT(MONTH from saledate)='11' THEN saledate
END)) AS Nov_numdays,
SUM(CASE WHEN EXTRACT(MONTH from saledate)=12 THEN amt END) AS December,
COUNT(DISTINCT(CASE WHEN EXTRACT(MONTH from saledate)='12' THEN saledate
END)) AS Dec_numdays,
((December/Dec_numdays)-
(November/Nov_numdays))/(November/Nov_numdays)*100 AS bump
FROM trnsact t JOIN strinfo ON t.store=strinfo.store
JOIN skuinfo si ON t.sku=si.sku
JOIN deptinfo d ON si.dept=d.dept
WHERE t.stype='p' AND t.store||EXTRACT(YEAR from t.saledate)||EXTRACT(MONTH
from t.saledate) IN
(SELECT store||EXTRACT(YEAR from saledate)||EXTRACT(MONTH from saledate)
FROM trnsact
GROUP BY store, EXTRACT(YEAR from saledate), EXTRACT(MONTH from saledate)
HAVING COUNT(DISTINCT saledate)>=20)
GROUP BY strinfo.store, strinfo.city, strinfo.state, d.deptdesc
HAVING November>1000 AND December>1000
ORDER BY bump DESC;
```

Results | SQL

| STORE | CITY | STATE | DEPTDESC | NOVEMBER | NOV_NUM... | DECEMBER | DEC_NUM... | BUMP |
|---|---|---|---|---|---|---|---|---|
| 3403 | SALINA | KS | LOUISVL | 1664.75 | 20 | 12170.25 | 21 | 596.00 |
| 9806 | MABELVALE | AR | FREDERI | 4399.48 | 24 | 24288.90 | 23 | 476.00 |
| 404 | PINE BLUFF | AR | MAI | 1170.93 | 23 | 8277.33 | 30 | 442.00 |
| 4502 | TALLAHASSEE | FL | ETERNIT | 2816.00 | 22 | 19270.50 | 28 | 438.00 |
| 9806 | MABELVALE | AR | P&Y | 18008.18 | 25 | 94996.48 | 25 | 428.00 |
| 8507 | TEXAS CITY | TX | MAI | 1019.00 | 27 | 5880.67 | 30 | 419.00 |
| 8802 | HAMMOND | LA | BLUE | 4607.00 | 27 | 26524.55 | 30 | 418.00 |
| 1607 | DALLAS | TX | LOUISVL | 4012.00 | 28 | 22212.00 | 30 | 417.00 |
| 3204 | SHREVEPORT | LA | BORA | 3443.25 | 26 | 19850.50 | 29 | 417.00 |
| 2807 | HARLINGEN | TX | BLUE | 7253.52 | 27 | 39631.86 | 29 | 409.00 |
| 7202 | MONTGOMERY | AL | ST JOHN | 5281.08 | 28 | 28739.93 | 30 | 408.00 |
| 3809 | HELENA | MT | ENVIRON | 27839.41 | 29 | 145627.18 | 30 | 406.00 |
| 709 | GLENDALE | AZ | BLUE | 12816.00 | 29 | 67123.45 | 30 | 406.00 |
| 1207 | MESQUITE | TX | BLUE | 11250.00 | 29 | 58498.56 | 30 | 403.00 |

Page 1 of 324 (16168 rows total)

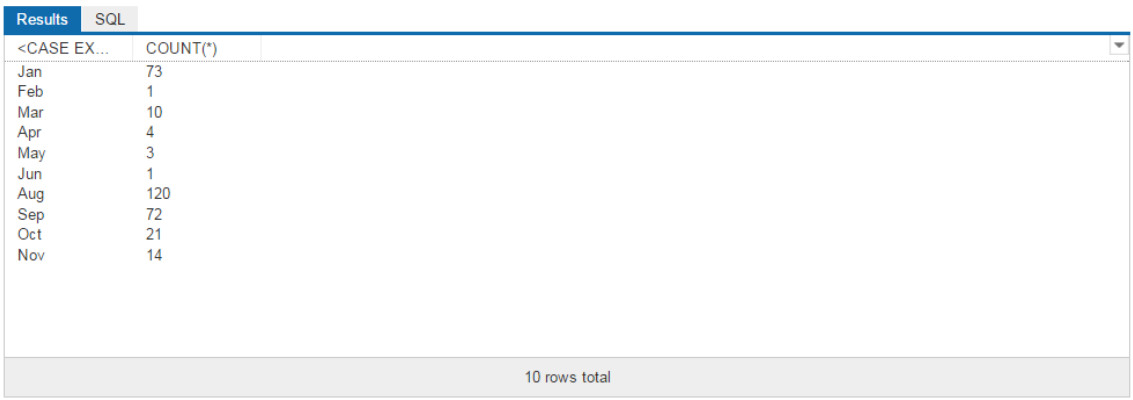
## Reliable Sales Trend

We want to examine whether monthly sales trends are consistent across stores.

- For each store, we identified the month with the minimum average daily revenue. For each of the twelve months of the year, we will count how many stores' minimum average daily revenue was in that month.
  The result showed that over 100 stores have their minimum average daily revenue in August.

```
SELECT CASE WHEN max_month_table.month_num=1 THEN 'Jan'
WHEN max_month_table.month_num=2 THEN 'Feb'
```

```
WHEN max_month_table.month_num=3 THEN 'Mar'
WHEN max_month_table.month_num=4 THEN 'Apr'
WHEN max_month_table.month_num=5 THEN 'May'
WHEN max_month_table.month_num=6 THEN 'Jun'
WHEN max_month_table.month_num=7 THEN 'Jul'
WHEN max_month_table.month_num=8 THEN 'Aug'
WHEN max_month_table.month_num=9 THEN 'Sep'
WHEN max_month_table.month_num=10 THEN 'Oct'
WHEN max_month_table.month_num=11 THEN 'Nov'
WHEN max_month_table.month_num=12 THEN 'Dec'
END, COUNT(*)
FROM
(SELECT DISTINCT EXTRACT(YEAR from saledate) AS year_num, EXTRACT(MONTH from
saledate) AS month_num,
CASE WHEN EXTRACT(YEAR from saledate)=2005 AND EXTRACT(MONTH from
saledate)=8 THEN 'exclude' END AS exclude_flag,
store, SUM(amt) AS tot_sales, COUNT(DISTINCT saledate) AS numdays,
tot_sales/numdays AS dailyrev, ROW_NUMBER() OVER (PARTITION BY store ORDER BY
dailyrev DESC) AS month_rank
FROM trnsact
WHERE stype='p' AND exclude_flag IS NULL AND store||EXTRACT(YEAR from
saledate)||EXTRACT(MONTH from saledate) IN
(SELECT store||EXTRACT(YEAR from saledate)||EXTRACT(MONTH from saledate) FROM
trnsact
GROUP BY store, EXTRACT(YEAR from saledate),EXTRACT(MONTH from saledate)
HAVING COUNT(DISTINCT saledate)>=20)
GROUP BY store, month_num, year_num
HAVING numdays>=20 QUALIFY month_rank=12) AS max_month_table
GROUP BY max_month_table.month_num
ORDER BY max_month_table.month_num;
```



Results | SQL

| <CASE EX... | COUNT(*) |
|---|---|
| Jan | 73 |
| Feb | 1 |
| Mar | 10 |
| Apr | 4 |
| May | 3 |
| Jun | 1 |
| Aug | 120 |
| Sep | 72 |
| Oct | 21 |
| Nov | 14 |

10 rows total

- Next we identified the month in which each store had its maximum number of SKU units returned. In December, the greatest number of stores have their maximum number of SKU units returned.

```
SELECT CASE WHEN max_month_table.month_num=1 THEN 'Jan'
WHEN max_month_table.month_num=2 THEN 'Feb'
WHEN max_month_table.month_num=3 THEN 'Mar'
WHEN max_month_table.month_num=4 THEN 'Apr'
WHEN max_month_table.month_num=5 THEN 'May'
WHEN max_month_table.month_num=6 THEN 'Jun'
WHEN max_month_table.month_num=7 THEN 'Jul'
WHEN max_month_table.month_num=8 THEN 'Aug'
WHEN max_month_table.month_num=9 THEN 'Sep'
WHEN max_month_table.month_num=10 THEN 'Oct'
WHEN max_month_table.month_num=11 THEN 'Nov'
WHEN max_month_table.month_num=12 THEN 'Dec'
END, COUNT(*)
FROM
(SELECT DISTINCT EXTRACT(YEAR from saledate) AS year_num, EXTRACT(MONTH from
saledate) AS month_num,
CASE WHEN EXTRACT(YEAR from saledate)=2005 AND EXTRACT(MONTH from
saledate)=8 THEN 'exclude' END AS exclude_flag,
store, SUM(quantity) AS tot_returns, ROW_NUMBER() OVER (PARTITION BY store
ORDER BY tot_returns DESC) AS month_rank
FROM trnsact
WHERE stype='r' AND exclude_flag IS NULL AND store||EXTRACT(YEAR from
saledate)||EXTRACT(MONTH from saledate) IN
(SELECT store||EXTRACT(YEAR from saledate)||EXTRACT(MONTH from saledate) FROM
trnsact
GROUP BY store, EXTRACT(YEAR from saledate),EXTRACT(MONTH from saledate)
HAVING COUNT(DISTINCT saledate)>=20)
GROUP BY store, month_num, year_num
QUALIFY month_rank=1) AS max_month_table
GROUP BY max_month_table.month_num
ORDER BY max_month_table.month_num;
```

Results | SQL

| <CASE EX... | COUNT(*) |
|---|---|
| Jan | 2 |
| Feb | 9 |
| Mar | 10 |
| Jul | 9 |
| Aug | 1 |
| Nov | 1 |
| Dec | 296 |