SQL Individual Project_ Bo(Barry) Huang

Introduction

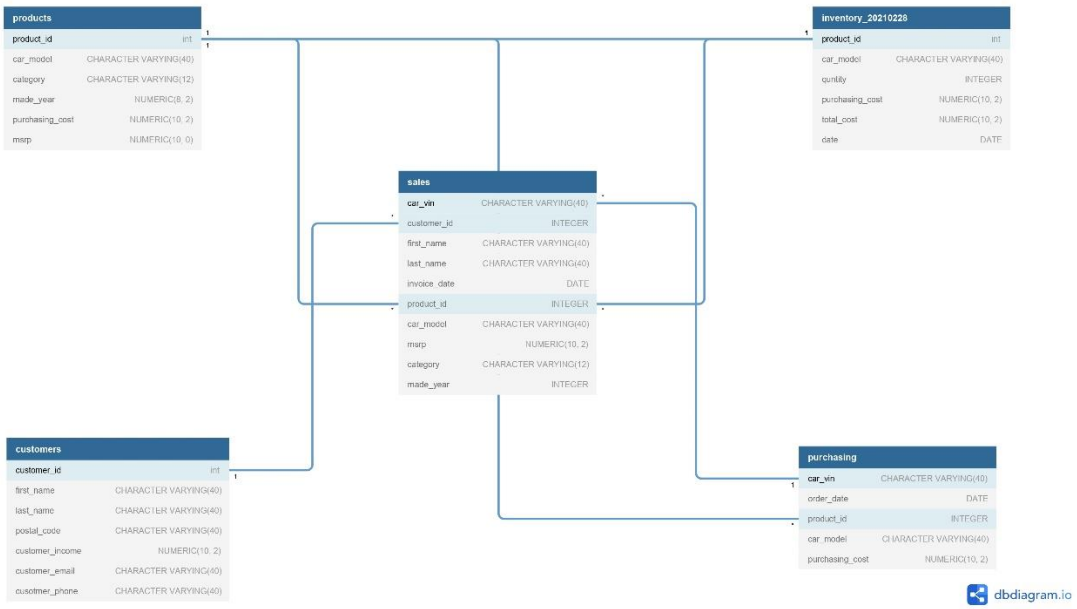
This analysis is about sales of a virtual Toyota dealer in Waterloo, Ontario from Mar. to May, 2021. All info is mocked data generated on https://mockaroo.com/. I will call this dealer as ABC Toyota.

Data-set

The database SQL_Individual_project has 5 tables as below:

- Customers-contains detail info related the customers, such as customer_id, first_name, last_name, postal_code, customer_income, customer_email and customer_phone;
- Inventory_20210228-contains the inventory info on Feb. 28, 2021 of ABC Toyota, such as product_id, car_model, quantity, unit_purchasing_price, total_cost and date;
- Products-contains detail info related products, such as product_id, car_model, category, made_year, purchasing_cost, and mrsp(selling price)
- Purchasing-contains info regarding ABC Toyota purchased new inventory in these 3 months, such as car_vin, order_date, product_id, car_model, and purchasing_cost;
- Sales-contains info related all sales in these 3 months, including car_vin, customer_id, first_name, last_name, invoice_date, product_id, car_model, msrp, category, and made_year.

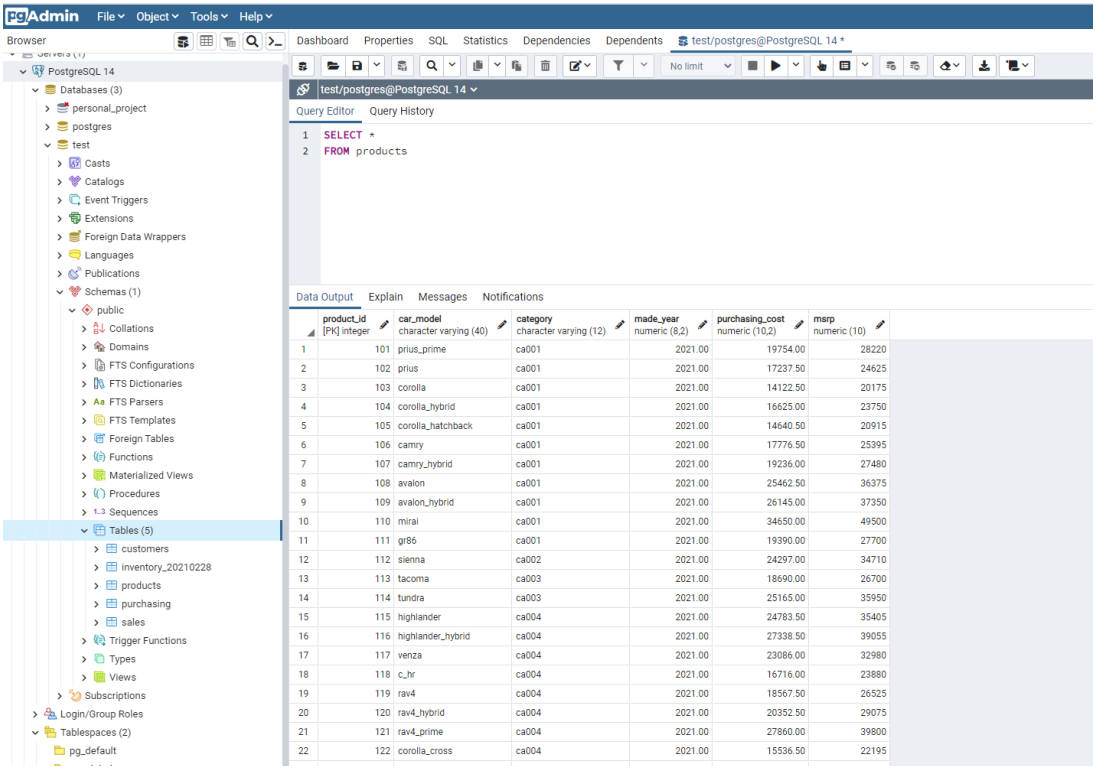
Here are the relationships among all tables(Schema):



Goals & Analysis

First of all, I created all tables and input all data in the database by the following queries:

```
--create table named products
CREATE TABLE products (
        product_id SERIAL PRIMARY KEY,
        car_model CHARACTER VARYING (40) NOT NULL,
        category CHARACTER VARYING (12),
        made_year NUMERIC(8,2),
        purchasing_cost NUMERIC(10,2) NOT NULL,
        msrp NUMERIC(10,0) NOT NULL
        );
-- copy data from csv file to table products
COPY products
FROM 'D:\Business Analytics\8075 SQL\Individual Project\CSV\products.csv' WITH CSV HEADER;
```

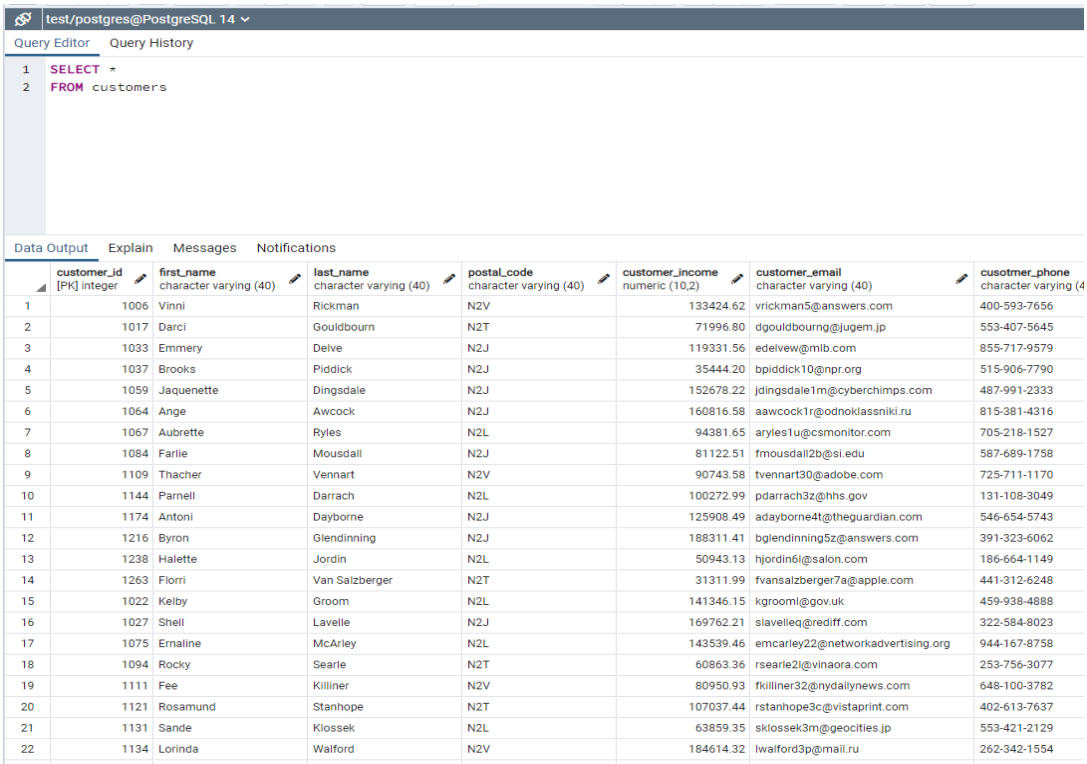


--create table named customers

```
CREATE TABLE customers (
        customer_id SERIAL PRIMARY KEY,
        first_name CHARACTER VARYING (40) NOT NULL,
        last_name CHARACTER VARYING (40) NOT NULL,
        postal_code CHARACTER VARYING (40) NOT NULL,
        customer_income NUMERIC(10,2) NOT NULL,
        customer_email CHARACTER VARYING (40) NOT NULL,
        cusotmer_phone CHARACTER VARYING (40) NOT NULL
                );
```

-- copy data from csv file to table customers

```
COPY customers
FROM 'D:\Business Analytics\8075 SQL\Individual Project\CSV\customers.csv' WITH CSV HEADER;
```



test/postgres@PostgreSQL 14

Query Editor | Query History

```
SELECT *
FROM customers
```

Data Output | Explain | Messages | Notifications

	customer_id [PK] integer	first_name character varying (40)	last_name character varying (40)	postal_code character varying (40)	customer_income numeric (10,2)	customer_email character varying (40)	cusotmer_phone character varying (4
1	1006	Vinni	Rickman	N2V	133424.62	vrickman5@answers.com	400-593-7656
2	1017	Darci	Gouldbourn	N2T	71996.80	dgouldbourng@jugem.jp	553-407-5645
3	1033	Emmery	Delve	N2J	119331.56	edelvew@mlb.com	855-717-9579
4	1037	Brooks	Piddick	N2J	35444.20	bpiddick10@npr.org	515-906-7790
5	1059	Jaquenette	Dingsdale	N2J	152678.22	jdingsdale1m@cyberchimps.com	487-991-2333
6	1064	Ange	Awcock	N2J	160816.58	aawcock1r@odnoklassniki.ru	815-381-4316
7	1067	Aubrette	Ryles	N2L	94381.65	aryles1u@csmonitor.com	705-218-1527
8	1084	Farlie	Mousdall	N2J	81122.51	fmousdall2b@si.edu	587-689-1758
9	1109	Thacher	Vennart	N2V	90743.58	tvennart30@adobe.com	725-711-1170
10	1144	Parnell	Darrach	N2L	100272.99	pdarrach3z@hhs.gov	131-108-3049
11	1174	Antoni	Dayborne	N2J	125908.49	adayborne4t@theguardian.com	546-654-5743
12	1216	Byron	Glendinning	N2J	188311.41	bglendinning5z@answers.com	391-323-6062
13	1238	Halette	Jordin	N2L	50943.13	hjordin6l@salon.com	186-664-1149
14	1263	Florri	Van Salzberger	N2T	31311.99	fvansalzberger7a@apple.com	441-312-6248
15	1022	Kelby	Groom	N2L	141346.15	kgrooml@gov.uk	459-938-4888
16	1027	Shell	Lavelle	N2J	169762.21	slavelleq@rediff.com	322-584-8023
17	1075	Ernaline	McArley	N2L	143539.46	emcarley22@networkadvertising.org	944-167-8758
18	1094	Rocky	Searle	N2T	60863.36	rsearle2l@vinaora.com	253-756-3077
19	1111	Fee	Killiner	N2V	80950.93	fkilliner32@nydailynews.com	648-100-3782
20	1121	Rosamund	Stanhope	N2T	107037.44	rstanhope3c@vistaprint.com	402-613-7637
21	1131	Sande	Klossek	N2L	63859.35	sklossek3m@geocities.jp	553-421-2129
22	1134	Lorinda	Walford	N2V	184614.32	lwalford3p@mail.ru	262-342-1554

--create table named purchasing

```
CREATE TABLE purchasing (
        car_vin CHARACTER VARYING (40) PRIMARY KEY,
        order_date DATE  NOT NULL,
        product_id INTEGER NOT NULL,
        car_model CHARACTER VARYING (40) NOT NULL,
        purchasing_cost NUMERIC(10,2) NOT NULL,
        FOREIGN KEY (product_id) REFERENCES products (product_id) ON UPDATE CASCADE ON DELETE CASCADE
        );
```

-- copy data from csv file to table purchasing

```
COPY purchasing
FROM 'D:\Business Analytics\8075 SQL\Individual Project\CSV\purchasing.csv' WITH CSV HEADER;
```

Data Output | Explain | Messages | Notifications

	car_vin [PK] character varying (40)	order_date date	product_id integer	car_model character varying (40)	purchasing_cost numeric (10,2)
1	WAULC58E95A785019	2021-01-02	123	4runner	26323.50
2	WAUJFAFHXBN016067	2021-01-02	123	4runner	26323.50
3	2T1BPRHE2EC564349	2021-01-02	123	4runner	26323.50
4	1GD01ZCGXEF923161	2021-01-02	123	4runner	26323.50
5	2T1BU4EE4DC381436	2021-01-02	123	4runner	26323.50
6	JM1DE1KY5D0190640	2021-01-02	123	4runner	26323.50
7	WAUNF78P97A303391	2021-01-02	123	4runner	26323.50
8	WBAYM1C55ED025896	2021-01-02	123	4runner	26323.50
9	WBSBL93442J890124	2021-01-02	123	4runner	26323.50
10	WUAW2BFC8FN499128	2021-01-02	123	4runner	26323.50
11	WA1VMAFP0FA560294	2021-01-02	123	4runner	26323.50
12	WBAPH5C53AA429251	2021-01-02	123	4runner	26323.50
13	WAU3GAFC7DN740649	2021-01-02	123	4runner	26323.50
14	JN8AZ2NCXC9132986	2021-01-02	123	4runner	26323.50
15	4JGDF2EE3FA020371	2021-01-02	108	avalon	25462.50
16	2HNYD18673H259259	2021-01-02	108	avalon	25462.50
17	WBA3B5C56EF931129	2021-01-02	108	avalon	25462.50
18	5N1CR2MM1EC852192	2021-01-02	108	avalon	25462.50
19	1GKKRNEDXCJ621434	2021-01-02	108	avalon	25462.50
20	WAUEG94FX6N852479	2021-01-02	108	avalon	25462.50
21	WA1DGAFE5BD386975	2021-01-02	108	avalon	25462.50

```
--create table named inventory_20210228

CREATE TABLE inventory_20210228 (

        product_id SERIAL PRIMARY KEY,

        car_model CHARACTER VARYING (40) NOT NULL,

        quntity INTEGER,

        purchasing_cost NUMERIC(10,2) NOT NULL,

        total_cost NUMERIC(10,2) NOT NULL,

        date DATE NOT NULL,

        FOREIGN KEY (product_id) REFERENCES products (product_id) ON UPDATE CASCADE ON DELETE CASCADE

        --FOREIGN KEY (product_id) REFERENCES purchasing (product_id) ON UPDATE CASCADE ON DELETE CASCADE

        );

-- copy data from csv file to table inventory_20210228

COPY inventory_20210228

FROM 'D:\Business Analytics\8075 SQL\Individual Project\CSV\inventory_20210228.csv' WITH CSV HEADER;
```

	product_id [PK] integer	car_model character varying (40)	quntity integer	purchasing_cost numeric (10,2)	total_cost numeric (10,2)	date date
1	101	prius_prime	11	19754.00	217294.00	2021-02-28
2	102	prius	6	17237.50	103425.00	2021-02-28
3	103	corolla	12	14122.50	169470.00	2021-02-28
4	104	corolla_hybrid	15	16625.00	249375.00	2021-02-28
5	105	corolla_hatchback	20	14640.50	292810.00	2021-02-28
6	106	camry	14	17776.50	248871.00	2021-02-28
7	107	camry_hybrid	2	19236.00	38472.00	2021-02-28
8	108	avalon	12	25462.50	305550.00	2021-02-28
9	109	avalon_hybrid	13	26145.00	339885.00	2021-02-28
10	110	mirai	11	34650.00	381150.00	2021-02-28
11	111	gr86	18	19390.00	349020.00	2021-02-28
12	112	sienna	10	24297.00	242970.00	2021-02-28
13	113	tacoma	1	18690.00	18690.00	2021-02-28
14	114	tundra	18	25165.00	452970.00	2021-02-28
15	115	highlander	17	24783.50	421319.50	2021-02-28
16	116	highlander_hybrid	16	27338.50	437416.00	2021-02-28
17	117	venza	3	23086.00	69258.00	2021-02-28
18	118	c_hr	4	16716.00	66864.00	2021-02-28
19	119	rav4	6	18567.50	111405.00	2021-02-28
20	120	rav4_hybrid	19	20352.50	386697.50	2021-02-28
21	121	rav4_prime	10	27860.00	278600.00	2021-02-28
22	122	corolla_cross	7	15536.50	108755.50	2021-02-28
23	123	4runner	2	26323.50	52647.00	2021-02-28

```
--create table named sales

CREATE TABLE sales (

        car_vin CHARACTER VARYING (40) PRIMARY KEY,

        customer_id INTEGER NOT NULL,

        first_name CHARACTER VARYING (40) NOT NULL,

        last_name CHARACTER VARYING (40) NOT NULL,

        invoice_date DATE NOT NULL,

        product_id INTEGER NOT NULL,

        car_model CHARACTER VARYING (40) NOT NULL,

        msrp NUMERIC(10,2) NOT NULL,

        category CHARACTER VARYING (12),

        made_year INTEGER,

        FOREIGN KEY (product_id) REFERENCES products (product_id) ON UPDATE CASCADE ON DELETE CASCADE,

        FOREIGN KEY (customer_id) REFERENCES customers (customer_id) ON UPDATE CASCADE ON DELETE CASCADE,

        FOREIGN KEY (car_vin) REFERENCES purchasing (car_vin) ON UPDATE CASCADE ON DELETE CASCADE,

        FOREIGN KEY (product_id) REFERENCES inventory_20210228 (product_id) ON UPDATE CASCADE ON DELETE CASCADE

        );

-- copy data from csv file to table sales

COPY sales

FROM 'D:\Business Analytics\8075 SQL\Individual Project\CSV\sales.csv' WITH CSV HEADER;
```

	car_vin [PK] character varying (40)	customer_id integer	first_name character varying (40)	last_name character varying (40)	invoice_date date	product_id integer	car_model character varying (40)	msrp numeric (10,2)	category character varying (12)	made_year integer
1	WAULC58E95A785019	1006	Vinni	Rickman	2021-04-11	123	4runner	37605.00	ca004	2021
2	WAUJFAFHXBN016067	1017	Darci	Gouldbourn	2021-03-24	123	4runner	37605.00	ca004	2021
3	2T1BPRHE2EC564349	1033	Emmery	Delve	2021-03-01	123	4runner	37605.00	ca004	2021
4	1GD01ZCGXEF923161	1037	Brooks	Piddick	2021-04-13	123	4runner	37605.00	ca004	2021
5	2T1BU4EE4DC381436	1059	Jaquenette	Dingsdale	2021-03-07	123	4runner	37605.00	ca004	2021
6	JM1DE1KY5D0190640	1064	Ange	Awcock	2021-05-07	123	4runner	37605.00	ca004	2021
7	WAUNF78P97A303391	1067	Aubrette	Ryles	2021-04-29	123	4runner	37605.00	ca004	2021
8	WBAYM1C55ED025896	1084	Farlie	Mousdall	2021-05-30	123	4runner	37605.00	ca004	2021
9	WBSBL93442J890124	1109	Thacher	Vennart	2021-05-14	123	4runner	37605.00	ca004	2021
10	WUAW2BFC8FN499128	1144	Parnell	Darrach	2021-04-24	123	4runner	37605.00	ca004	2021
11	WA1VMAFP0FA560294	1174	Antoni	Dayborne	2021-04-17	123	4runner	37605.00	ca004	2021

After creating those tables, I did the following analysis:

1. I used the following query to check how many models per category to get a basic idea of products sold by this dealer. Other product info could be found on table products.

--Check the number of models for each category:

SELECT category, COUNT(car_model) AS category_count

FROM products

GROUP BY category

	category character varying (12)	category_count bigint
1	ca001	11
2	ca002	1
3	ca004	10
4	ca003	2

2. Check sales numbers by car models in last 3 months, and sort the sales numbers decrescent. At the same time, I create a view for this result for sales team to check easily. By this query, sales team can easily know which model is the most popular one in last three months.

SELECT car_model, COUNT(car_model) AS model_sales_count

FROM sales

GROUP BY car_model

ORDER BY model_sales_count DESC

--Check sales numbers by car models, sort the sales numbers decrescent (View):

CREATE VIEW sales_car_model AS

SELECT car_model, COUNT(car_model) AS model_sales_count

FROM sales

GROUP BY car_model

ORDER BY model_sales_count DESC

	car_model character varying (40)	model_sales_count bigint
1	tundra	18
2	tacoma	18
3	highlander	18
4	rav4_prime	16
5	highlander_hybrid	15
6	corolla	14
7	avalon	14
8	4runner	14
9	camry	13
10	corolla_cross	12
11	mirai	12
12	corolla_hatchback	12
13	venza	12
14	c_hr	11
15	sequoia	11
16	gr86	11
17	camry_hybrid	11
18	avalon_hybrid	11
19	sienna	10
20	prius_prime	10
21	rav4	10
22	rav4_hybrid	9
23	corolla_hybrid	9

3. For financial & sales departments, it is important to know the original inventory in order to do the comparison. The following query is to check the total inventory cost on Mar.

```
SELECT SUM(A.total_cost) AS inventory_cost, SUM(B.purchasing_cost)AS
total_purchasing_cost, (SUM(A.total_cost)+SUM(B.purchasing_cost)) AS total_inventory_Mar

FROM inventory_20210228 A

INNER JOIN purchasing B

ON A.product_id=B.product_id
```

	inventory_cost numeric	total_purchasing_cost numeric	total_inventory_mar numeric
1	74491532.50	6699427.00	81190959.50

4. For sales team, the customers who bought the most expensive product might be treated carefully since these customers could be good customers for some other products. The following query is to check customers' info who bought the most expensive car in the last three months:

```
SELECT customer_id, first_name, last_name, invoice_date, car_model, msrp

FROM sales

WHERE msrp=(

                    SELECT MAX(msrp)

                    FROM sales)
```

	customer_id integer	first_name character varying (40)	last_name character varying (40)	invoice_date date	car_model character varying (40)	msrp numeric (10,2)
1	1052	Farlee	Cosins	2021-04-07	sequoia	50500.00
2	1053	Massimiliano	Samples	2021-04-04	sequoia	50500.00
3	1077	Joice	Caush	2021-05-18	sequoia	50500.00
4	1107	Cristian	Vanyatin	2021-03-19	sequoia	50500.00
5	1116	Rowena	Middle	2021-03-22	sequoia	50500.00
6	1125	Elysee	Trayhorn	2021-04-24	sequoia	50500.00
7	1180	Avril	Scriviner	2021-03-01	sequoia	50500.00
8	1183	Traci	Shatliff	2021-04-10	sequoia	50500.00
9	1187	Lotti	Sivess	2021-04-18	sequoia	50500.00
10	1245	Leontine	Tuttle	2021-05-29	sequoia	50500.00
11	1290	Joye	Goding	2021-05-01	sequoia	50500.00

5. As a company, profit is always one of the most important KPI's. Based on the dataset, I used the following query to check the margin to present profit for each sales at the end of May for last three months. This table is very important for sales manager. By this table, the sales manager could know margins for each sales(the margin for each sales should be vary in the real world.), then he could help him to do some marketing activities, such as promotion on some specific models. So I also create a view for the result of this query for the sales manager.

SELECT A.first_name, A.last_name, A.invoice_date, A.product_id, A.car_model, A.msrp, B.purchasing_cost, (A.msrp-B.purchasing_cost) AS margin

FROM sales A

INNER JOIN products B

ON A.product_id = B. product_id

--Create view of margin by customers: margin_by_customer

CREATE VIEW margin_by_customer AS

SELECT A.first_name, A.last_name, A.invoice_date, A.product_id, A.car_model, A.msrp, B.purchasing_cost, (A.msrp-B.purchasing_cost) AS margin

FROM sales A

INNER JOIN products B

ON A.product_id = B. product_id

	first_name character varying (40)	last_name character varying (40)	invoice_date date	product_id integer	car_model character varying (40)	msrp numeric (10,2)	purchasing_cost numeric (10,2)	margin numeric
1	Vinni	Rickman	2021-04-11	123	4runner	37605.00	26323.50	11281.50
2	Darci	Gouldbourn	2021-03-24	123	4runner	37605.00	26323.50	11281.50
3	Emmery	Delve	2021-03-01	123	4runner	37605.00	26323.50	11281.50
4	Brooks	Piddick	2021-04-13	123	4runner	37605.00	26323.50	11281.50
5	Jaquenette	Dingsdale	2021-03-07	123	4runner	37605.00	26323.50	11281.50
6	Ange	Awcock	2021-05-07	123	4runner	37605.00	26323.50	11281.50
7	Aubrette	Ryles	2021-04-29	123	4runner	37605.00	26323.50	11281.50
8	Farlie	Mousdall	2021-05-30	123	4runner	37605.00	26323.50	11281.50
9	Thacher	Vennart	2021-05-14	123	4runner	37605.00	26323.50	11281.50
10	Parnell	Darrach	2021-04-24	123	4runner	37605.00	26323.50	11281.50
11	Antoni	Dayborne	2021-04-17	123	4runner	37605.00	26323.50	11281.50
12	Byron	Glendinning	2021-04-11	123	4runner	37605.00	26323.50	11281.50
13	Halette	Jordin	2021-03-22	123	4runner	37605.00	26323.50	11281.50
14	Florri	Van Salzberger	2021-05-11	123	4runner	37605.00	26323.50	11281.50
15	Kelby	Groom	2021-05-27	108	avalon	36375.00	25462.50	10912.50
16	Shell	Lavelle	2021-03-18	108	avalon	36375.00	25462.50	10912.50
17	Ernaline	McArley	2021-05-08	108	avalon	36375.00	25462.50	10912.50
18	Rocky	Searle	2021-03-16	108	avalon	36375.00	25462.50	10912.50
19	Fee	Killiner	2021-04-04	108	avalon	36375.00	25462.50	10912.50
20	Rosamund	Stanhope	2021-03-23	108	avalon	36375.00	25462.50	10912.50
21	Sande	Klossek	2021-05-19	108	avalon	36375.00	25462.50	10912.50

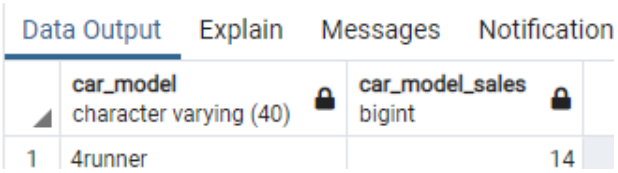
6. Different area should have different customers' profile. If the marketing and sales team could have the clear customer's profiles in specific area, they could have some specific marketing activities in specific area, then the dealer might have better sales and cost savings. The following query is to check the income level of all customers in different areas of Waterloo

```
SELECT postal_code,
CASE WHEN customer_income>80000 THEN 'HIGH'
        WHEN customer_income BETWEEN 50000 AND 79999 THEN 'MEDIUM'
        ELSE 'LOW'
END AS customer_level,
COUNT (DISTINCT customer_id) AS customer_count
FROM customers
GROUP BY postal_code,customer_level
ORDER BY customer_level
```

	postal_code character varying (40)	customer_level text	customer_count bigint
1	N2J	HIGH	64
2	N2L	HIGH	53
3	N2T	HIGH	48
4	N2V	HIGH	50
5	N2J	LOW	10
6	N2L	LOW	7
7	N2T	LOW	7
8	N2V	LOW	3
9	N2J	MEDIUM	13
10	N2L	MEDIUM	16
11	N2T	MEDIUM	13
12	N2V	MEDIUM	16

7. Regarding the sales person, sometimes it is also important to know the sales situation for specific models in the time period, then he could do more efforts on some specific models based on his customer base. The following query is to check the total sold number of specific model , such as “4 Runner”, in last three months:

```
SELECT  car_model, COUNT(car_model) AS car_model_sales

FROM sales

WHERE car_model='4runner'

GROUP BY car_model
```

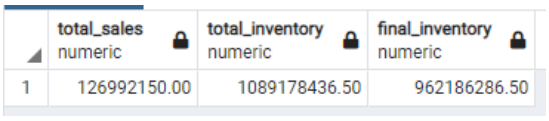


Data Output | Explain | Messages | Notification

	car_model character varying (40)	car_model_sales bigint
1	4runner	14

8. For financial department, inventory is always important to control the company's profit and cost. The following query is to check the total inventory cost on May 31. Since it is so important, I also create a view for it, then sales manager could also check it easily.

```
SELECT SUM(A.msrp) AS total_sales, (SUM (B.purchasing_cost)+SUM(C.total_cost)) AS total_inventory, ((SUM (B.purchasing_cost)+SUM(C.total_cost))-SUM(A.msrp)) AS final_inventory

FROM sales A

INNER JOIN purchasing B  ON A.product_id=B.product_id

INNER JOIN inventory_20210228 C ON A.product_id=C.product_id

--Create view of final inventory cost on May 31, 2021:inventory_20210531

CREATE VIEW final_inventory_20210531 AS

SELECT SUM(A.msrp) AS total_sales, (SUM (B.purchasing_cost)+SUM(C.total_cost)) AS total_inventory, ((SUM (B.purchasing_cost)+SUM(C.total_cost))-SUM(A.msrp)) AS final_inventory

FROM sales A  INNER JOIN purchasing B  ON A.product_id=B.product_id

INNER JOIN inventory_20210228 C ON A.product_id=C.product_id
```



	total_sales numeric	total_inventory numeric	final_inventory numeric
1	126992150.00	1089178436.50	962186286.50

9. For sales team, it is always important to know the sales situation in some specific time period. If there are some patterns could be found, the sales team might have better sales revenue in the following period. The following query is to check and sort sales in specific time period by models. The time period could be changed to get any specific time period.

SELECT car_model, SUM(msrp) AS sales_sum

FROM sales

WHERE invoice_date BETWEEN '20210401' AND '20210430'

GROUP BY car_model

ORDER BY sales_sum DESC

	car_model character varying (40)	sales_sum numeric
1	tundra	359500.00
2	rav4_prime	278600.00
3	sequoia	252500.00
4	4runner	225630.00
5	highlander	212430.00
6	avalon_hybrid	186750.00
7	avalon	181875.00
8	venza	164900.00
9	camry_hybrid	164880.00
10	tacoma	160200.00
11	highlander_hybrid	156220.00
12	camry	152370.00
13	mirai	148500.00
14	c_hr	143280.00
15	corolla	121050.00
16	corolla_hybrid	118750.00
17	prius_prime	112880.00
18	corolla_cross	110975.00
19	gr86	110800.00
20	prius	98500.00
21	rav4_hybrid	58150.00
22	rav4	53050.00

10. Sales by area is another important info for sales team. It might present people in some areas might like to buy Toyota more than other areas. The following query is to check sales by postal codes(areas).

In order to be checked by sales team, it is good to create a view for this query as well.

SELECT A.postal_code, SUM(B.msrp) AS area_sales

FROM customers A

INNER JOIN sales B

ON A.customer_id = B.customer_id

GROUP BY A.postal_code

--Create view of sales by postal codes(areas):

```
CREATE VIEW sales_by_area AS
SELECT A.postal_code, SUM(B.msrp) AS area_sales
FROM customers A
INNER JOIN sales B
ON  A.customer_id = B.customer_id
GROUP BY A.postal_code
```

	postal_code character varying (40)	area_sales numeric
1	N2J	2948270.00
2	N2T	2082855.00
3	N2L	2404485.00
4	N2V	2135000.00

Since not all people could reach all info easily, it is also very important to create some views which have important info. People could check these views easily. Here are two more views created:

1. Views for customers income situation by areas can help sales people to understand customers in different areas would afford different models with different prices.

```
--Create View: customer_income_level
CREATE VIEW customer_income_level AS
SELECT postal_code,
CASE WHEN customer_income>80000 THEN 'HIGH'
        WHEN customer_income BETWEEN 50000 AND 79999 THEN 'MEDIUM'
        ELSE 'LOW'
```

END AS customer_level,

COUNT (DISTINCT customer_id) AS customer_count

FROM customers

GROUP BY postal_code,customer_level

ORDER BY customer_level

	postal_code character varying (40)	customer_level text	customer_count bigint
1	N2J	HIGH	64
2	N2L	HIGH	53
3	N2T	HIGH	48
4	N2V	HIGH	50
5	N2J	LOW	10
6	N2L	LOW	7
7	N2T	LOW	7
8	N2V	LOW	3
9	N2J	MEDIUM	13
10	N2L	MEDIUM	16
11	N2T	MEDIUM	13
12	N2V	MEDIUM	16

2, View of full customer info contains all detailed info of the customers in the past 3 months. If the sales people want to contact or follow any specific customers, they can check the info in this view.

CREATE VIEW customer_full_info AS

SELECT sal.car_vin, sal.product_id, sal.car_model, sal.category, sal.made_year, sal.msrp, sal.customer_id, sal.first_name, sal.last_name, cus.customer_income, cus.customer_email, cus.cusotmer_phone, cus.postal_code

FROM sales sal

INNER JOIN customers cus

ON sal.customer_id=cus.customer_id

	car_vin character varying (40)	product_id integer	car_model character varying (40)	category character varying (12)	made_year integer	msrp numeric (10,2)	customer_id integer	first_name character varying (40)	last_name character varying (40)	customer_income numeric (10,2)	customer_email character varying (40)
1	WAULC58E95A785019	123	4runner	ca004	2021	37605.00	1006	Vinni	Rickman	133424.62	vrickman5@answers.co
2	WAUJFAFHXBN016067	123	4runner	ca004	2021	37605.00	1017	Darci	Gouldbourn	71996.80	dgouldbourng@jugem.j
3	2T1BPRHE2EC564349	123	4runner	ca004	2021	37605.00	1033	Emmery	Delve	119331.56	edelvew@mlb.com
4	1GD01ZCGXEF923161	123	4runner	ca004	2021	37605.00	1037	Brooks	Piddick	35444.20	bpiddick10@npr.org
5	2T1BU4EE4DC381436	123	4runner	ca004	2021	37605.00	1059	Jaquenette	Dingsdale	152678.22	jdingsdale1m@cyberch
6	JM1DE1KY5D0190640	123	4runner	ca004	2021	37605.00	1064	Ange	Awcock	160816.58	aawcock1r@odnoklass
7	WAUNF78P97A303391	123	4runner	ca004	2021	37605.00	1067	Aubrette	Ryles	94381.65	aryles1u@csmonitor.co
8	WBAYM1C55ED025896	123	4runner	ca004	2021	37605.00	1084	Farlie	Mousdall	81122.51	fmousdall2b@si.edu
9	WBSBL93442J890124	123	4runner	ca004	2021	37605.00	1109	Thacher	Vennart	90743.58	tvennart30@adobe.com
10	WUAW2BFC8FN499128	123	4runner	ca004	2021	37605.00	1144	Parnell	Darrach	100272.99	pdarrach3z@hhs.gov
11	WA1VMAFP0FA560294	123	4runner	ca004	2021	37605.00	1174	Antoni	Dayborne	125908.49	adayborne4t@theguard
12	WBAPH5C53AA429251	123	4runner	ca004	2021	37605.00	1216	Byron	Glendinning	188311.41	bglendinning5z@answe
13	WAU3GAFC7DN740649	123	4runner	ca004	2021	37605.00	1238	Halette	Jordin	50943.13	hjordin6i@salon.com
14	JN8AZ2NCXC9132986	123	4runner	ca004	2021	37605.00	1263	Florri	Van Salzberger	31311.99	fvansalzberger7a@appl
15	4JGDF2EE3FA020371	108	avalon	ca001	2021	36375.00	1022	Kelby	Groom	141346.15	kgroom|@gov.uk
16	2HNYD18673H259259	108	avalon	ca001	2021	36375.00	1027	Shell	Lavelle	169762.21	slavelleq@rediff.com
17	WBA3B5C56EF931129	108	avalon	ca001	2021	36375.00	1075	Ernaline	McArley	143539.46	emcarley22@networka
18	5N1CR2MM1EC852192	108	avalon	ca001	2021	36375.00	1094	Rocky	Searle	60863.36	rsearle2l@vinaora.com
19	1GKKRNEDXCJ621434	108	avalon	ca001	2021	36375.00	1111	Fee	Killiner	80950.93	fkilliner32@nydailynew
20	WAUEG94FX6N852479	108	avalon	ca001	2021	36375.00	1121	Rosamund	Stanhope	107037.44	rstanhope3c@vistaprin
21	WA1DGAFE5BD386975	108	avalon	ca001	2021	36375.00	1131	Sande	Klossek	63859.35	sklossek3m@geocities

Conclusion:

Based on those analysis above, I can get the following conclusions:

1. The inventory was increased by the end of the period, but the numbers of vehicles didn't increase. It means the dealer didn't sell the expensive models well. The sales team should pay more attention on selling the expensive models or decrease the inventory of expensive models.
2. The areas with higher income level customers bought more cars, so those areas should do more marketing activities to get even better results. At the same time, sales team should put more efforts on selling expensive models on these areas.
3. Pickup truck has the highest sales number, and the margin of trucks are very good, so the dealer should get more inventory of trucks, and marketing and sales team should put more efforts on these two models of truck for better profit.

Data source: https://mockaroo.com/