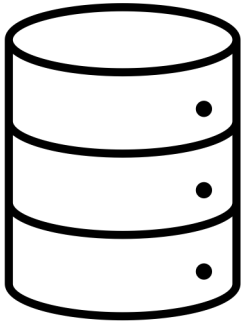
Data Warehousing Project

- Retail Sales -

CS779 Advanced Data Management

Aug 16, 2022



Boston University Metropolitan College, Kazuya DOHI

BOSTON UNIVERSITY

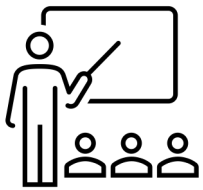
Agenda

- Introduction & Project overview
- Workflow
- Data organization
- Data loading
- Schema Design (Normalized / Dimensional)
- ETL
- Data visualization (for business questions)

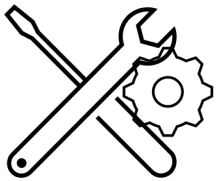
Introduction

Kazuya Dohi

Background
- Retail
- Supply Chain Management
- 10 years experience & MBA
- Tokyo, Shanghai, Hong Kong
- Second last class in MSADA



Why this course/program?
- To become a business translator



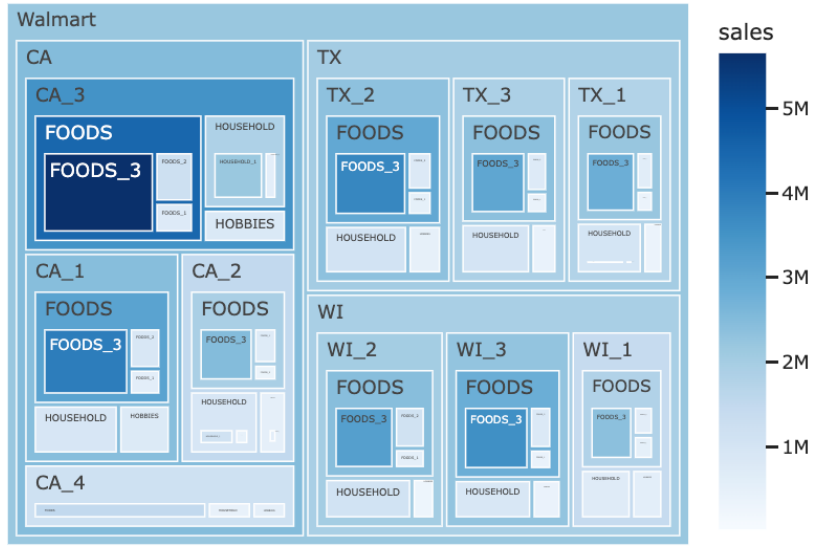
Python
SQL
Tableau

*Started programming since Sep 2021

BOSTON UNIVERSITY

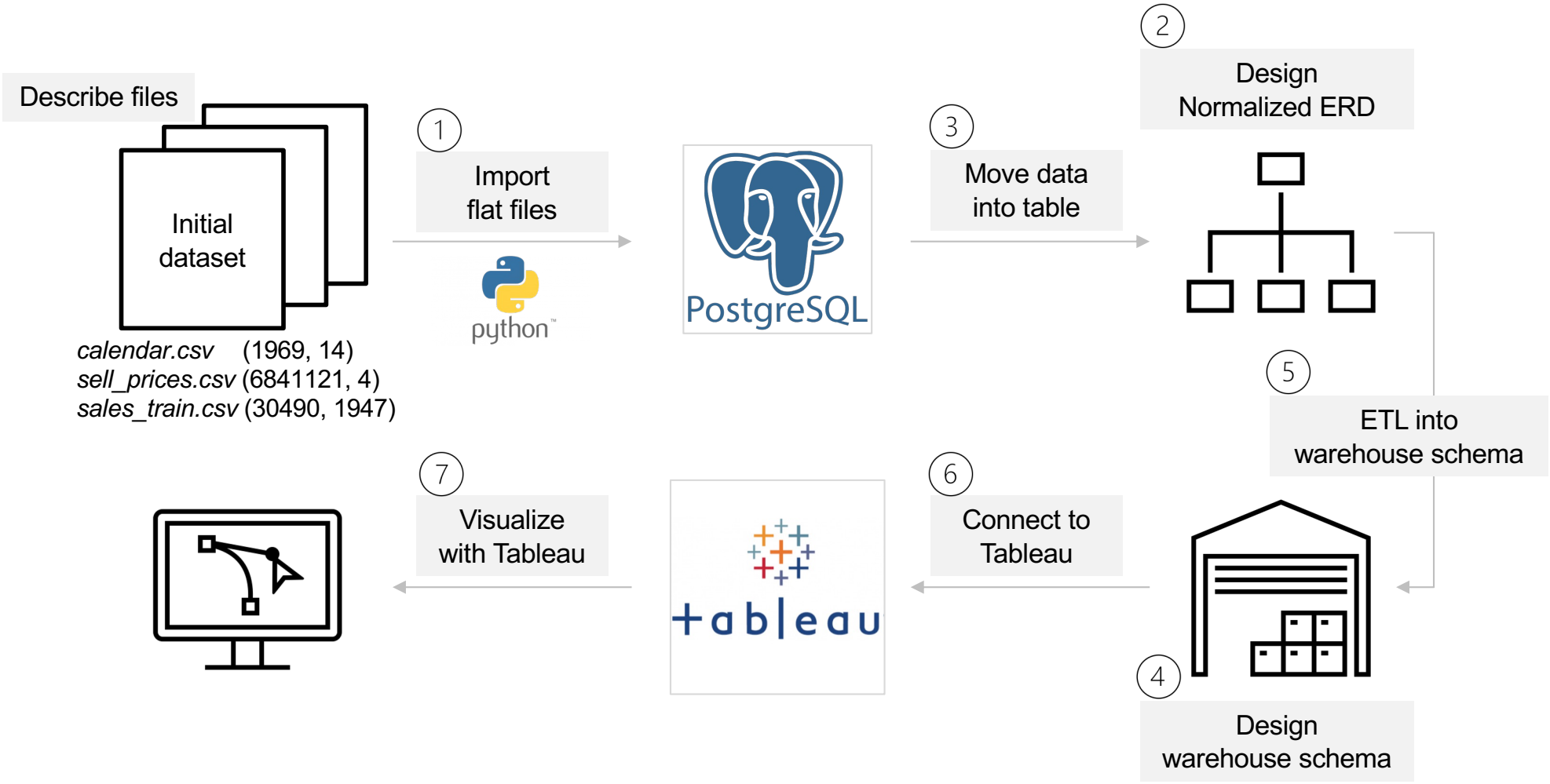
Project Overview

A data warehouse project
- Data sets of Walmart sales
- Python data integration
- Normalized/dimensional Schema design
- ETL
- Tableau visualization



BOSTON UNIVERSITY

Work Frow



BOSTON UNIVERSITY

Step 0

Describe data files

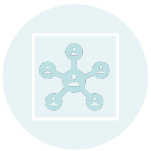
Import flat files into PostgreSQL

Design normalized ERD

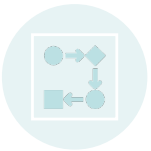
Write queries to move data from flat files to normalized data tables

Design data warehouse schema

ETL data from normalized schema into the warehouse schema



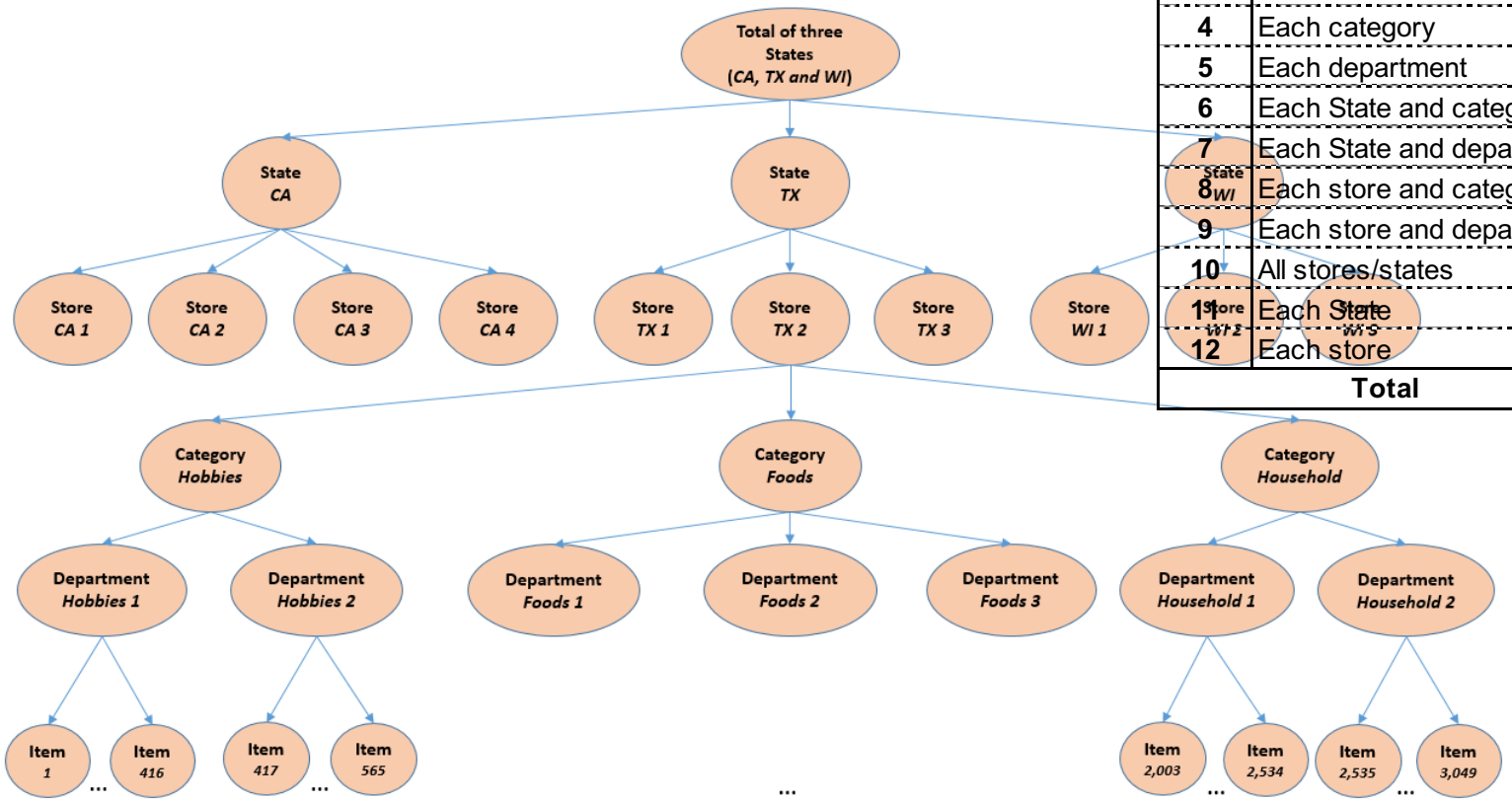
Connect to Tableau



Visualize with Tableau

Data Organization

Level id	Aggregation Level (Unit sales)	Product	Number of series
1	All stores/states	All	1
2	Each State	All	3
3	Each store	All	10
4	Each category	All	3
5	Each department	All	7
6	Each State and category	All	9
7	Each State and department	All	21
8	Each store and category	All	30
9	Each store and department	All	70
10	All stores/states	Single	3,049
11	Each State	Single	9,147
12	Each store	Single	30,490
Total			**42,840**



BOSTON UNIVERSITY

Data Files - 1

File 1: calendar.csv About the dates the products are sold

date	The date in a “y-m-d” format.
wm_yr_wk	The id of the week the date belongs to.
weekday	The type of the day (Saturday, Sunday, …, Friday)
wday	The id of the weekday, starting from Saturday
month	The month of the date.
year	The year of the date
event_name_1	If the date includes an event, the name of this event
event_type_1	If the date includes an event, the type of this event
event_name_2	If the date includes a second event, the name of this event
event_type_2	If the date includes a second event, the type of this event
snap_CA, TX, and WI	A binary variable indicating if the stores allow SNAP purchases

Size: (1969, 14)

	date	wm_yr_wk	weekday	wday	month	year	d	event_name_1	event_type_1	event_name_2	event_type_2	snap_CA	snap_TX	snap_WI
0	2011-01-29	11101	Saturday	1	1	2011	d_1	NaN	NaN	NaN	NaN	0	0	0
1	2011-01-30	11101	Sunday	2	1	2011	d_2	NaN	NaN	NaN	NaN	0	0	0
2	2011-01-31	11101	Monday	3	1	2011	d_3	NaN	NaN	NaN	NaN	0	0	0
3	2011-02-01	11101	Tuesday	4	2	2011	d_4	NaN	NaN	NaN	NaN	1	1	0
4	2011-02-02	11101	Wednesday	5	2	2011	d_5	NaN	NaN	NaN	NaN	1	0	1

Data Files - 2

File 2: sell_prices.csv About the price of the products sold per store and date

store_id	The id of the store where the product is sold
item_id	The id of the product
wm_yr_wk	The id of the week
sell_price	The price of the product for the given week/store.

Size: (6841121, 4)

	store_id	item_id	wm_yr_wk	sell_price
0	CA_1	HOBBIES_1_001	11325	9.58
1	CA_1	HOBBIES_1_001	11326	9.58
2	CA_1	HOBBIES_1_001	11327	8.26
3	CA_1	HOBBIES_1_001	11328	8.26
4	CA_1	HOBBIES_1_001	11329	8.26

Data Files - 3

File 3: sales_train.csv About the historical daily unit sales data per product and store

item_id	The id of the product
dept_id	The id of the department the product belongs to
cat_id	The id of the category the product belongs to
store_id	The id of the store where the product is sold
state_id	The State where the store is located
d_1, …, d_i, … d_1941	The number of units sold at day i, starting from 2011-01-29

Size: (30490, 1947)

Before

	id	item_id	dept_id	cat_id	store_id	state_id	d_1	d_2	d_3	d_4	...	d_1932	d_1933	d_1934	d_1935
0	HOBBIES_1_001_CA_1_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	2	4	0	0
1	HOBBIES_1_002_CA_1_evaluation	HOBBIES_1_002	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	0	1	2	1
2	HOBBIES_1_003_CA_1_evaluation	HOBBIES_1_003	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	1	0	2	0
3	HOBBIES_1_004_CA_1_evaluation	HOBBIES_1_004	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	1	1	0	4
4	HOBBIES_1_005_CA_1_evaluation	HOBBIES_1_005	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	0	0	0	2

After

	item_id	dept_id	cat_id	store_id	state_id	date_id	sales_quantity
10275130	HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_1	CA	338	0
10275131	HOBBIES_1_002	HOBBIES_1	HOBBIES	CA_1	CA	338	0
10275132	HOBBIES_1_003	HOBBIES_1	HOBBIES	CA_1	CA	338	0
10275133	HOBBIES_1_004	HOBBIES_1	HOBBIES	CA_1	CA	338	0
10275134	HOBBIES_1_005	HOBBIES_1	HOBBIES	CA_1	CA	338	1

Only year 2 data was used
Transformed wide form to long form

Step 1

Describe data files

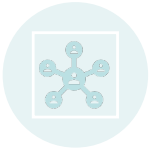
Import flat files into PostgreSQL

Design normalized ERD

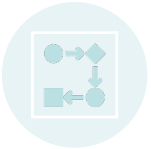
Write queries to move data from flat files to normalized data tables

Design data warehouse schema

ETL data from normalized schema into the warehouse schema



Connect to Tableau



Visualize with Tableau

Loading Files into Database

- Language: Python
- Library: psycopg2, sqlalchemy
- Database: PostgreSQL

```
def table_creation(sql_name, name):
    host = 'localhost'; database = 'CS779_Project_new';
    user = 'postgres'; password = 'password';
    port = '5432';

    conn = psycopg2.connect(
                host = host, database = database,
                user = user, password = password,
                port = port)
    cursor = conn.cursor()

    # drop table if it already exists, and create new table
    cursor.execute('drop table if exists ' + name)
    cursor.execute(sql_name)

    conn.commit()
    conn.close()

table_creation(create_table_calendar, 'Calendar')
table_creation(create_table_sells, 'Sells')
table_creation(create_table_price, 'SellPrice')
```

```
def table_load(load_data, name):
    conn_string =
    'postgresql://postgres:password@localhost:5432/CS779_Project_new'
    db = create_engine(conn_string)
    conn = db.connect()

    load_data.to_sql(name, con=conn, if_exists='append', index=False)

    conn = psycopg2.connect(conn_string)
    conn.autocommit = True
    cursor = conn.cursor()

    conn.commit()
    conn.close()

table_load(calendar, 'calendar')
table_load(sales, 'sells')
table_load(sell_prices, 'sellprice')
```

```
create_table_calendar = '''
CREATE TABLE Calendar(
date          timestamp(3) NOT NULL,
wm_yr_wk      numeric(5) NOT NULL,
weekday       varchar(9) NOT NULL,
wday          numeric(1) NOT NULL,
month         numeric(2) NOT NULL,
year          numeric(4) NOT NULL,
d             varchar(6) NOT NULL,
event_name_1  varchar(32),
event_type_1  varchar(32),
event_name_2  varchar(32),
event_type_2  varchar(32));
'''
create_table_sells = '''
CREATE TABLE Sells(
item_id        varchar(64) NOT NULL,
dept_id        varchar(64) NOT NULL,
cat_id         varchar(64) NOT NULL,
state_id       varchar(2) NOT NULL,
store_id       varchar(6) NOT NULL,
date_id              numeric(5) NOT NULL,
sales_quantity  numeric(4) NOT NULL);
'''
create_table_price = '''
CREATE TABLE SellPrice(
store_id       varchar(6) NOT NULL,
item_id        varchar(64) NOT NULL,
wm_yr_wk       numeric(5) NOT NULL,
sell_price     numeric(6,2) NOT NULL);
'''
```


BOSTON UNIVERSITY

Step 2

Describe data files

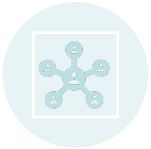
Import flat files into PostgreSQL

Design normalized ERD

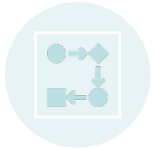
Write queries to move data from flat files to normalized data tables

Design data warehouse schema

ETL data from normalized schema into the warehouse schema

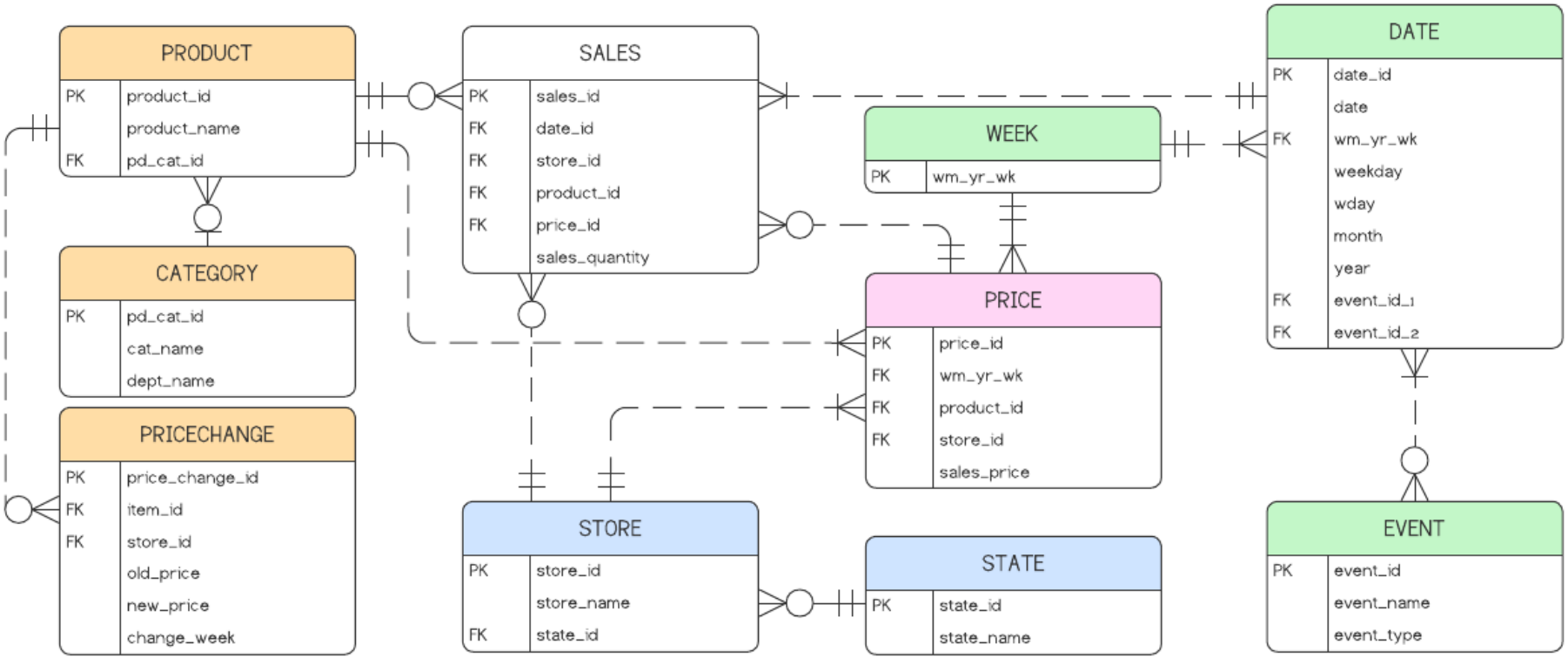


Connect to Tableau



Visualize with Tableau

Normalized Schema



BOSTON UNIVERSITY

Step 3

Describe data files

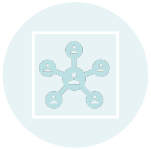
Import flat files into PostgreSQL

Design normalized ERD

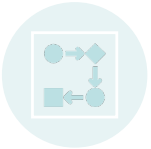
Write queries to move data from flat files to normalized data tables

Design data warehouse schema

ETL data from normalized schema into the warehouse schema



Connect to Tableau



Visualize with Tableau

Moving Data to Normalized Table


```sql
--Table creation
CREATE TABLE Event(
event_id     numeric(3) PRIMARY KEY,
event_name   varchar(64) NOT NULL,
event_type   varchar(64) NOT NULL);

CREATE TABLE Week(
wm_yr_wk      numeric(5) PRIMARY KEY);

CREATE TABLE Date(
date_id       numeric(5) PRIMARY KEY,
date          timestamp(3) NOT NULL,
wm_yr_wk      numeric(5) NOT NULL,
weekday       varchar(9) NOT NULL,
wday          numeric(1) NOT NULL,
month         numeric(2) NOT NULL,
year          numeric(4) NOT NULL,
event_id_1    numeric(3),
event_id_2    numeric(3),
CONSTRAINT fk_wm_yr_wk  FOREIGN KEY (wm_yr_wk) REFERENCES Week(wm_yr_wk),
CONSTRAINT fk_event_id_1 FOREIGN KEY (event_id_1) REFERENCES Event(event_id),
CONSTRAINT fk_event_id_2 FOREIGN KEY (event_id_2) REFERENCES Event(event_id));

--create sequence
CREATE SEQUENCE event_seq START WITH 1;
CREATE SEQUENCE date_seq START WITH 1;

--create indices useful when normalizing
CREATE INDEX calencar_date ON calendar(date);
CREATE INDEX calencar_wm_yr_wk ON calencar(wm_yr_wk);
```

```sql
--populate schemas
INSERT INTO event(event_id, event_name, event_type)
SELECT nextval('event_seq'), event_name_1, event_type_1
FROM (SELECT distinct event_name_1, event_type_1
   FROM Calendar WHERE event_name_1 IS NOT NULL) s;

INSERT INTO week(wm_yr_wk)
SELECT distinct wm_yr_wk
FROM Calendar ORDER BY wm_yr_wk;

INSERT INTO date
(date_id, date, wm_yr_wk, weekday, wday, month, year, event_id_1, event_id_2)
SELECT nextval('date_seq'),date, wm_yr_wk, weekday, wday, month, year, event_id_1, event_id
FROM
  (SELECT date, wm_yr_wk, weekday, wday, month, year, event_id_1, event_id
  FROM
  (SELECT date, wm_yr_wk, weekday, wday, month, year,
    event_id AS event_id_1, event_type_2 FROM calendar
  LEFT JOIN event on calendar.event_name_1 = event.event_name) s
  LEFT JOIN event on s.event_type_2 = event.event_name ORDER BY date) m;
```


BOSTON UNIVERSITY

Moving Data to Normalized Table

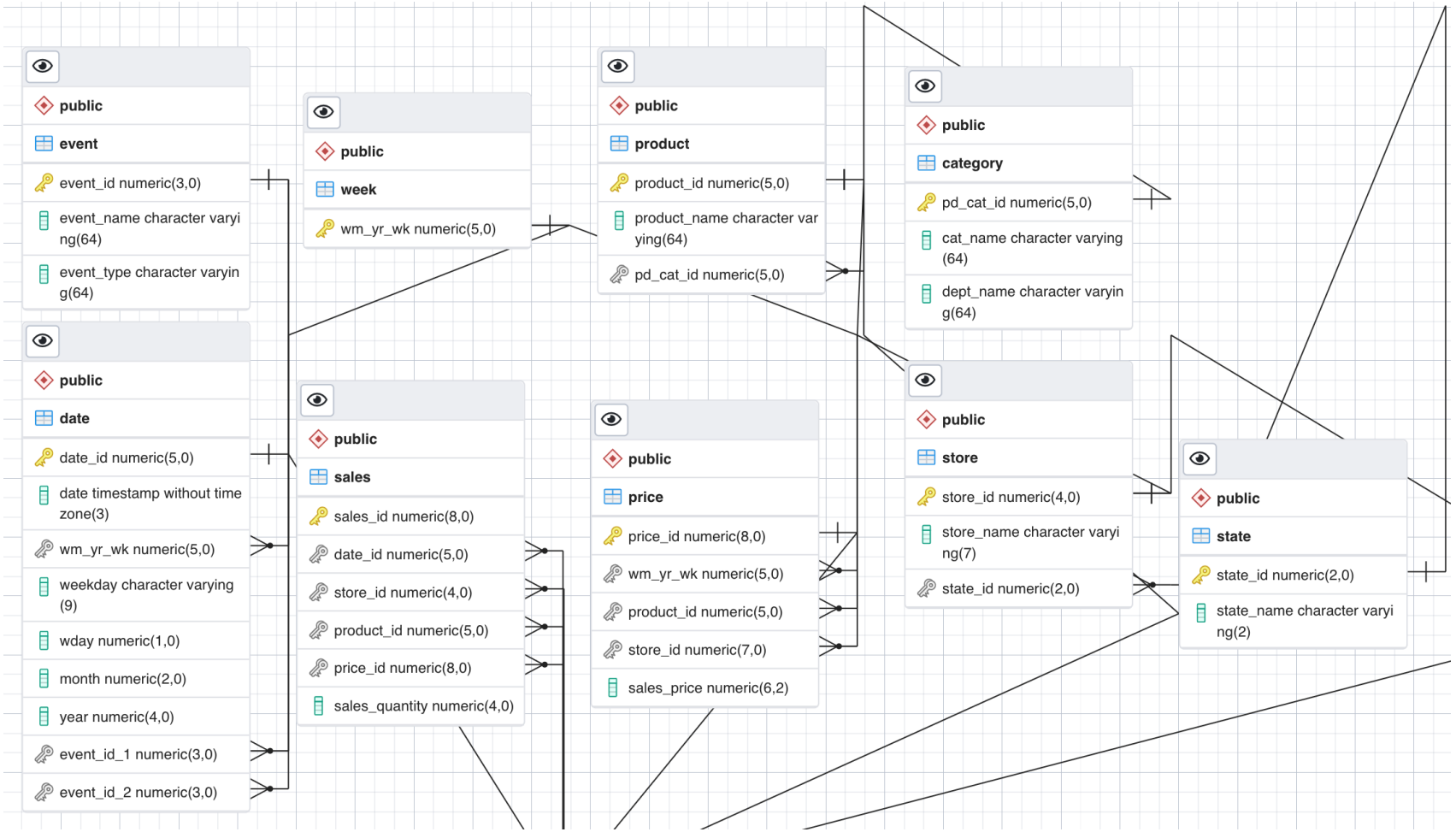

```sql
CREATE TABLE Sales(
sales_id       numeric(8) PRIMARY KEY,
date_id        numeric(5),
store_id       numeric(4),
product_id     numeric(5),
price_id       numeric(8) NOT NULL,
sales_quantity numeric(4) NOT NULL,
CONSTRAINT fk_date_id FOREIGN KEY (date_id) REFERENCES Date(date_id),
CONSTRAINT fk_store_id FOREIGN KEY (store_id) REFERENCES Store(store_id),
CONSTRAINT fk_product_id FOREIGN KEY (product_id) REFERENCES Product(product_id),
CONSTRAINT fk_price_id FOREIGN KEY (price_id) REFERENCES Price(price_id));

--create sequence
CREATE SEQUENCE sales_seq START WITH 1;

--create indices useful when normalizing
CREATE INDEX sells_item_id ON sells(item_id);
CREATE INDEX sells_dept_id ON sells(dept_id);
CREATE INDEX sells_cat_id ON sells(cat_id);
CREATE INDEX sells_state_id ON sells(state_id);
CREATE INDEX price_item_id ON price(item_id);
CREATE INDEX price_wm_yr_wk ON price(wm_yr_wk);
CREATE INDEX price_sell_price ON price(sell_price);
CREATE INDEX calencar_date ON calendar(date);
CREATE INDEX calencar_wm_yr_wk ON calencar(wm_yr_wk);
```

```sql
INSERT INTO Sales(sales_id, date_id, store_id, product_id, price_id, sales_quantity)
SELECT nextval('sales_seq'), m.date_id, store.store_id, product.product_id,
       price.price_id, m.sales_quantity
FROM product
  JOIN(
  SELECT item_id, store_id, s.wm_yr_wk, date.date_id, sales_quantity FROM sells
  LEFT JOIN (SELECT CAST(SUBSTRING(d, 3) AS numeric), wm_yr_wk, date FROM calendar) s
  ON sells.date_id = s.substring
  LEFT JOIN date ON s.date = date.date) m ON product.product_name = m.item_id
LEFT JOIN store ON m.store_id = store.store_name
LEFT JOIN price ON m.wm_yr_wk = price.wm_yr_wk
  AND store.store_id = price.store_id
  AND product.product_id = price.product_id
WHERE price_id IS NOT NULL
ORDER BY date_id, store_id, product_id;
```

	item_id	dept_id	cat_id	store_id	state_id	date_id	sales_quantity
10275130	HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_1	CA	338	0
10275131	HOBBIES_1_002	HOBBIES_1	HOBBIES	CA_1	CA	338	0
10275132	HOBBIES_1_003	HOBBIES_1	HOBBIES	CA_1	CA	338	0
10275133	HOBBIES_1_004	HOBBIES_1	HOBBIES	CA_1	CA	338	0
10275134	HOBBIES_1_005	HOBBIES_1	HOBBIES	CA_1	CA	338	1

BOSTON UNIVERSITY

Normalized Schema



BOSTON UNIVERSITY

Step 4

Describe data files

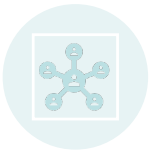
Import flat files into PostgreSQL

Design normalized ERD

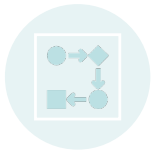
Write queries to move data from flat files to normalized data tables

Design data warehouse schema

ETL data from normalized schema into the warehouse schema



Connect to Tableau

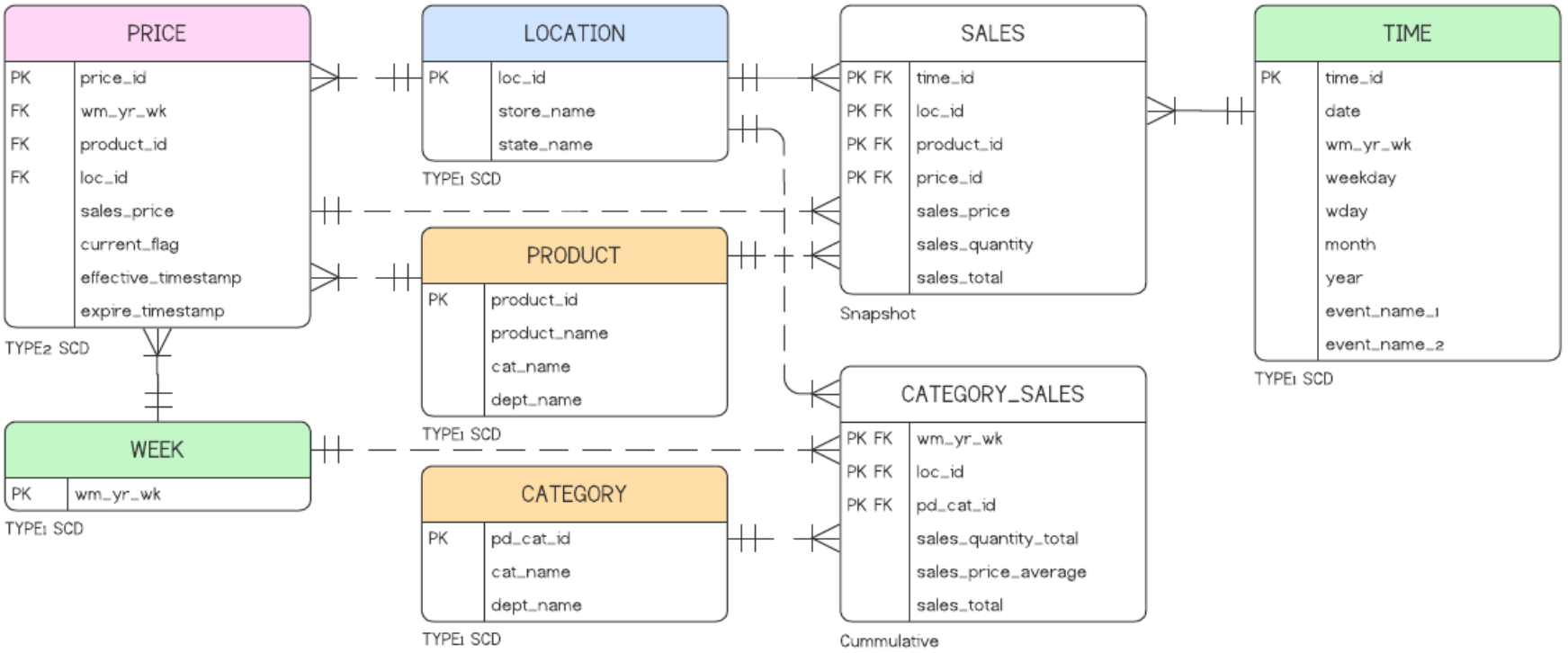


Visualize with Tableau

Business Questions

- Any difference in weekly categorical sales by store?
- Which department has the highest average price in a selected week?
- Did the weekly Hobbies_1 sale differ among stores in TX in a selected week?
- Which product has the largest sales quantity in California in a selected month?
- Which product has the largest sales in Texas in a selected week?

Warehouse Schema



BOSTON UNIVERSITY

Step 5

Describe data files

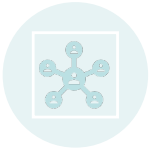
Import flat files into PostgreSQL

Design normalized ERD

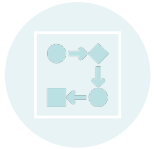
Write queries to move data from flat files to normalized data tables

Design data warehouse schema

ETL data from normalized schema into the warehouse schema

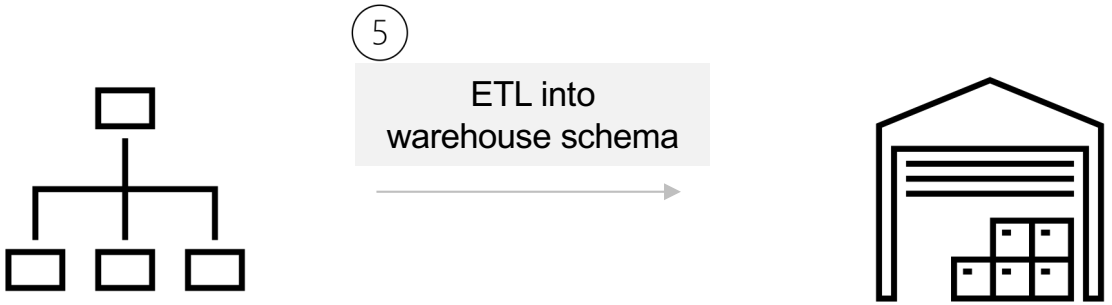


Connect to Tableau



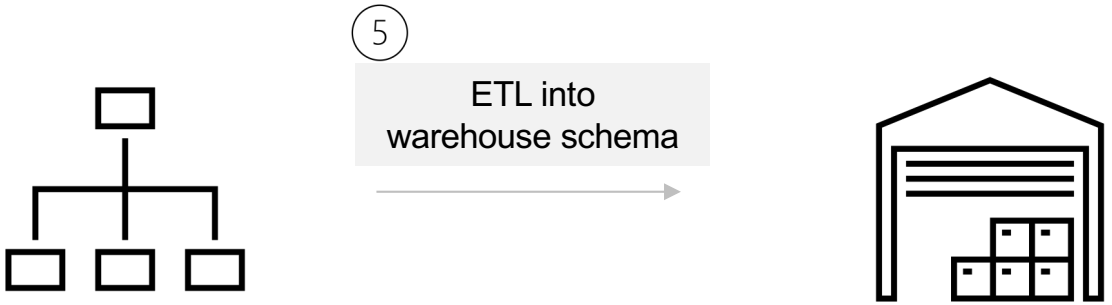
Visualize with Tableau

ETL to Warehouse Schema




```
INSERT INTO Sales
SELECT * FROM
dblink('host=localhost dbname=CS779_Project_new user=postgres password=password',
       'SELECT date_id, sales.store_id, sales.product_id, sales.price_id, sales_price, sales_quantity,
       sales_price*sales_quantity AS sales_total FROM sales
       LEFT JOIN price ON sales.price_id = price.price_id')
       AS t(time_id numeric, loc_id numeric, product_id numeric, price_id numeric, sales_price numeric,
         sales_quantity numeric, sales_total numeric);

INSERT INTO Category_Sales
SELECT * FROM
dblink('host=localhost dbname=CS779_Project_new user=postgres password=password',
       'SELECT s.wm_yr_wk, s.loc_id, s.pd_cat_id, s.sales_quantity_total,
       ROUND(AVG(s.sales_total/s.sales_quantity_total),2) AS sales_price_average, s.sales_total
       FROM (SELECT d.wm_yr_wk, sales.store_id AS loc_id, p.pd_cat_id, SUM(sales_quantity) AS sales_quantity_total,
             SUM(sales_price * sales_quantity) AS sales_total
             FROM sales
             LEFT JOIN (SELECT date_id, wm_yr_wk FROM date) d ON sales.date_id = d.date_id
             LEFT JOIN price ON sales.price_id = price.price_id
             LEFT JOIN (SELECT product_id, pd_cat_id FROM product) p ON sales.product_id = p.product_id
             GROUP BY d.wm_yr_wk, sales.store_id, p.pd_cat_id) s
       GROUP BY s.wm_yr_wk, s.loc_id, s.pd_cat_id, s.sales_quantity_total, s.sales_total')
       AS t(wm_yr_wk numeric, loc_id numeric, pd_cat_id numeric, sales_quantity_total numeric,
         sales_price_average numeric, sales_total numeric);
```

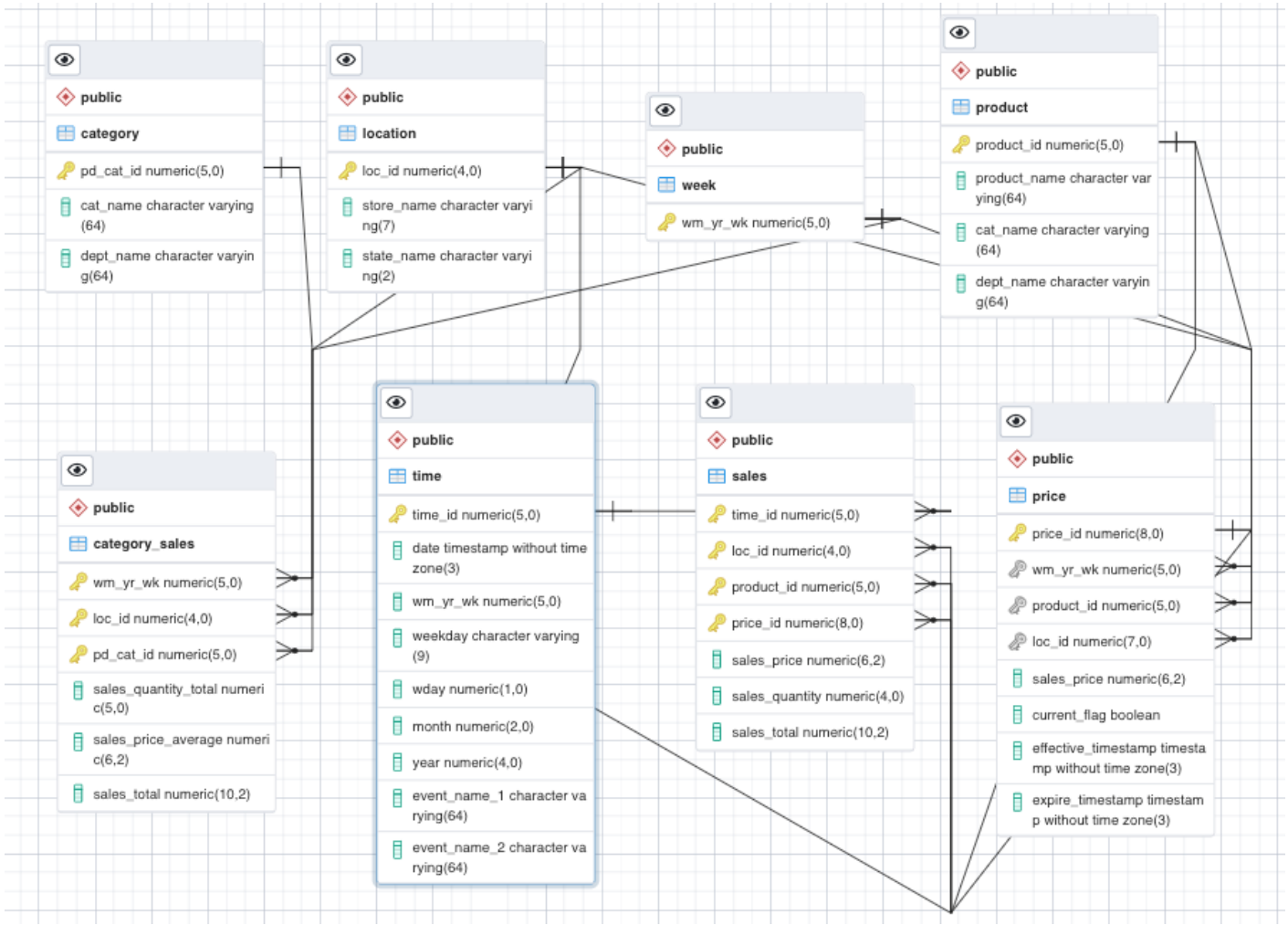
ETL to Warehouse Schema (in Python)




```
location = sales_train.loc[:, ['state_id', 'store_id']].drop_duplicates().reset_index(drop=True)
loc_id = pd.DataFrame({'loc_id': np.arange(1, location.shape[0] + 1)})
location = pd.concat([loc_id, location], axis=1)
location
```

	loc_id	state_id	store_id
0	1	CA	CA_1
1	2	CA	CA_2
2	3	CA	CA_3
3	4	CA	CA_4
4	5	TX	TX_1
5	6	TX	TX_2
6	7	TX	TX_3
7	8	WI	WI_1
8	9	WI	WI_2
9	10	WI	WI_3

Warehouse Schema



BOSTON UNIVERSITY

Step 6 & 7

Describe data files

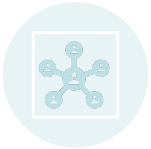
Import flat files into PostgreSQL

Design normalized ERD

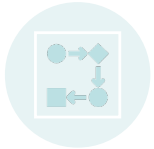
Write queries to move data from flat files to normalized data tables

Design data warehouse schema

ETL data from normalized schema into the warehouse schema



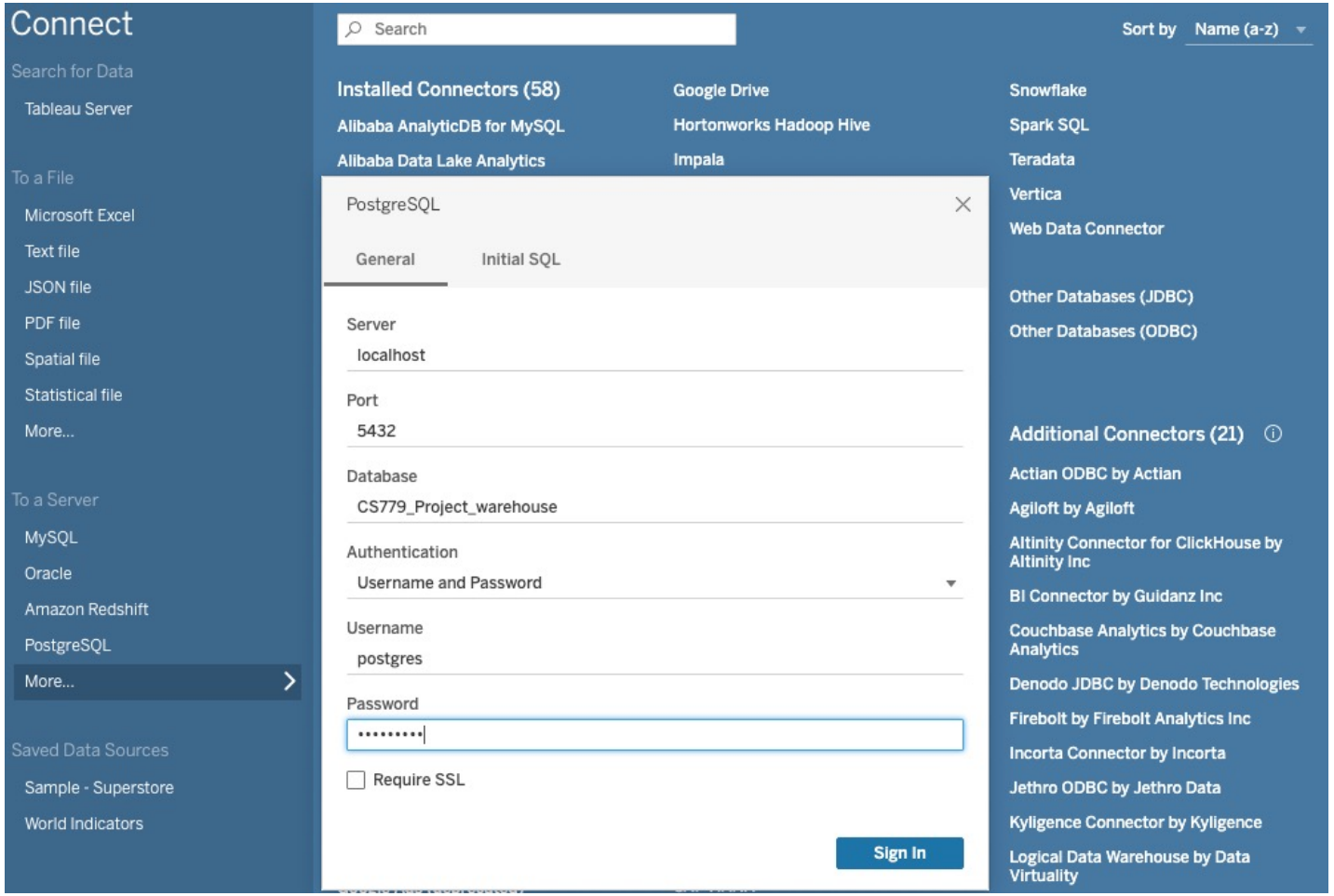
Connect to Tableau



Visualize with Tableau

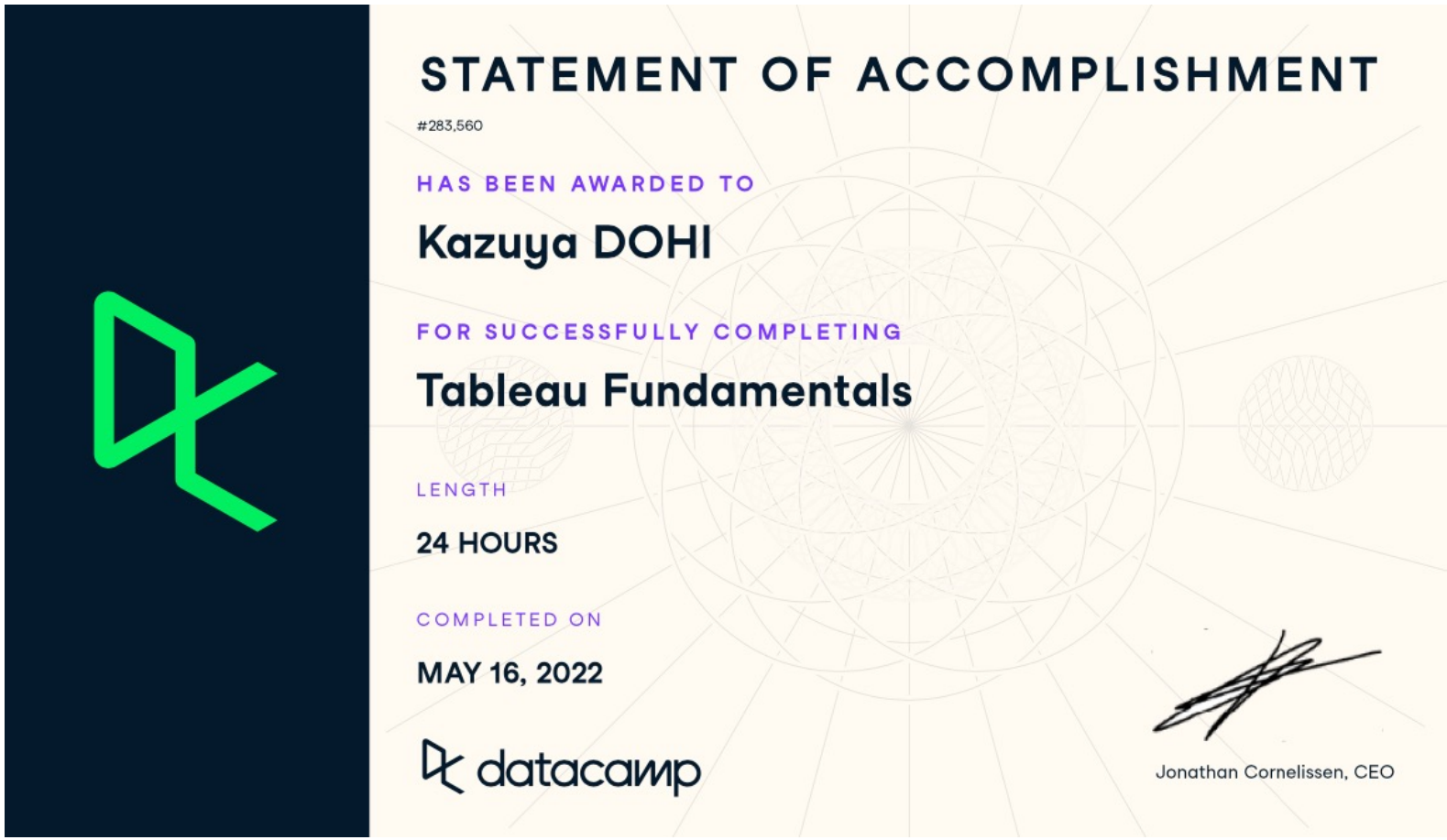
BOSTON UNIVERSITY

Connection to Tableau



BOSTON UNIVERSITY

Tableau Practice on Datacamp



STATEMENT OF ACCOMPLISHMENT

#283,560

HAS BEEN AWARDED TO

Kazuya DOHI

FOR SUCCESSFULLY COMPLETING

Tableau Fundamentals

LENGTH

24 HOURS

COMPLETED ON

MAY 16, 2022

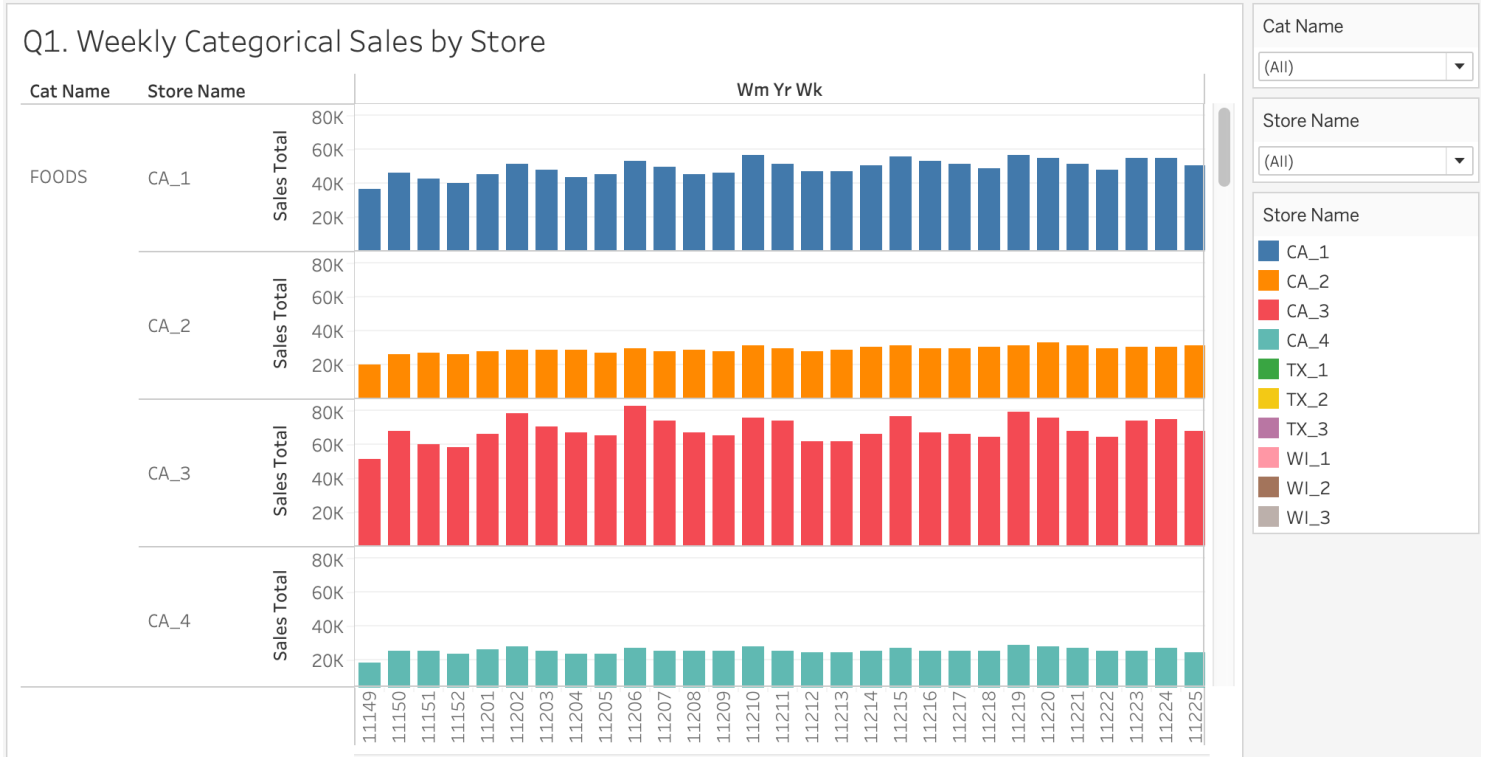
datacamp

Jonathan Cornelissen, CEO

BOSTON UNIVERSITY

Data Visualization in Tableau

Q1. Any difference in weekly categorical sales by store?

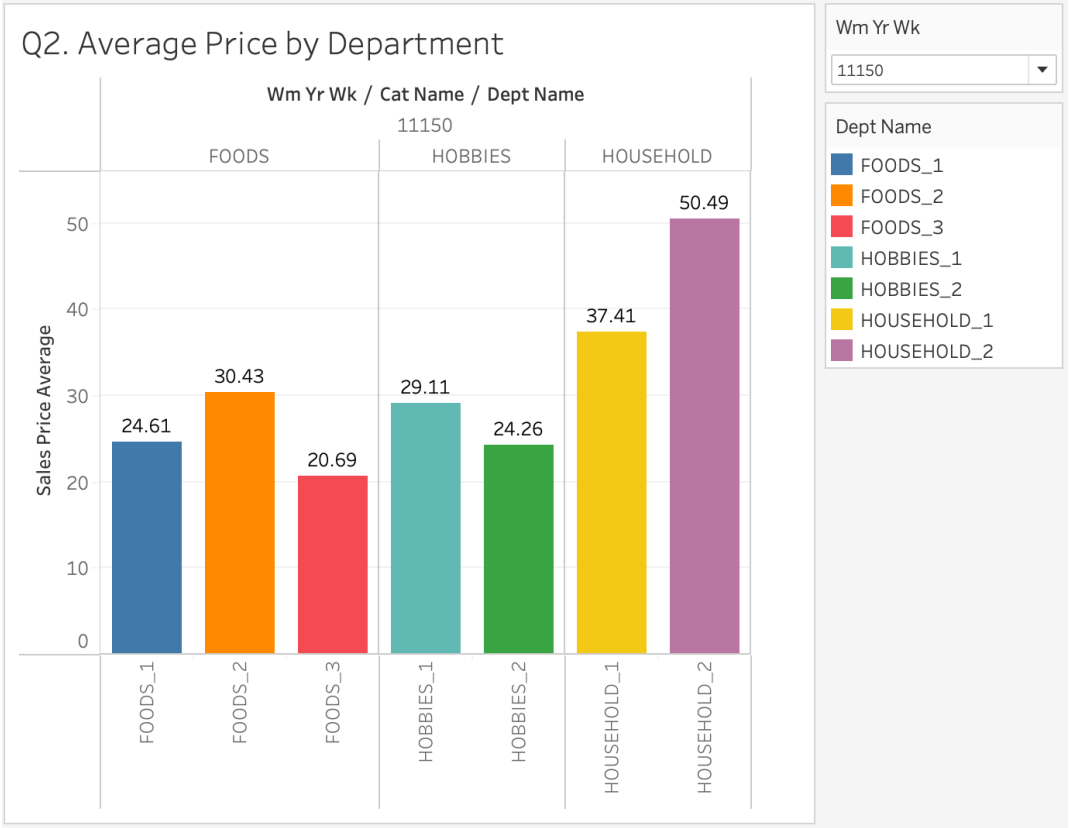


Store CA_1 has the highest sale among stores in CA.
CA_2 and 4 have a stable sale over the year but the sales of CA_1 and 3 fluctuated.

BOSTON UNIVERSITY

Data Visualization in Tableau

Q2. Which department has the highest average price in a selected week?



Household_2 has the highest average price in the week 11150.

*11150 -> 2011, week 50 (FY starts from the last week on Jan)

BOSTON UNIVERSITY

Data Visualization in Tableau

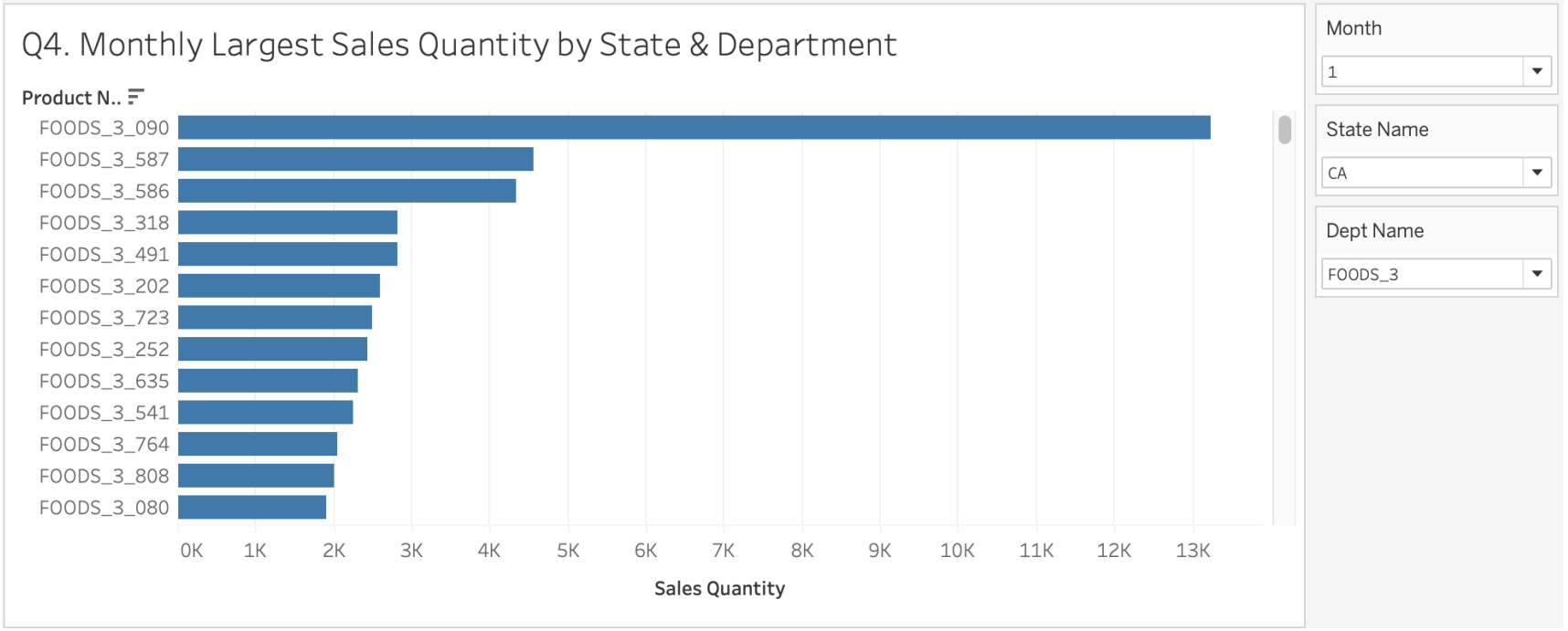
Q3. Did the weekly sales of department Hobbies_1 differ among stores in TX in a selected week?

TX_2 had the highest weekly sales of $4,863 among 3 stores in TX.
50 % higher than other stores.

BOSTON UNIVERSITY

Data Visualization in Tableau

Q4. Which product has the largest monthly sales quantity in a selected state and department?

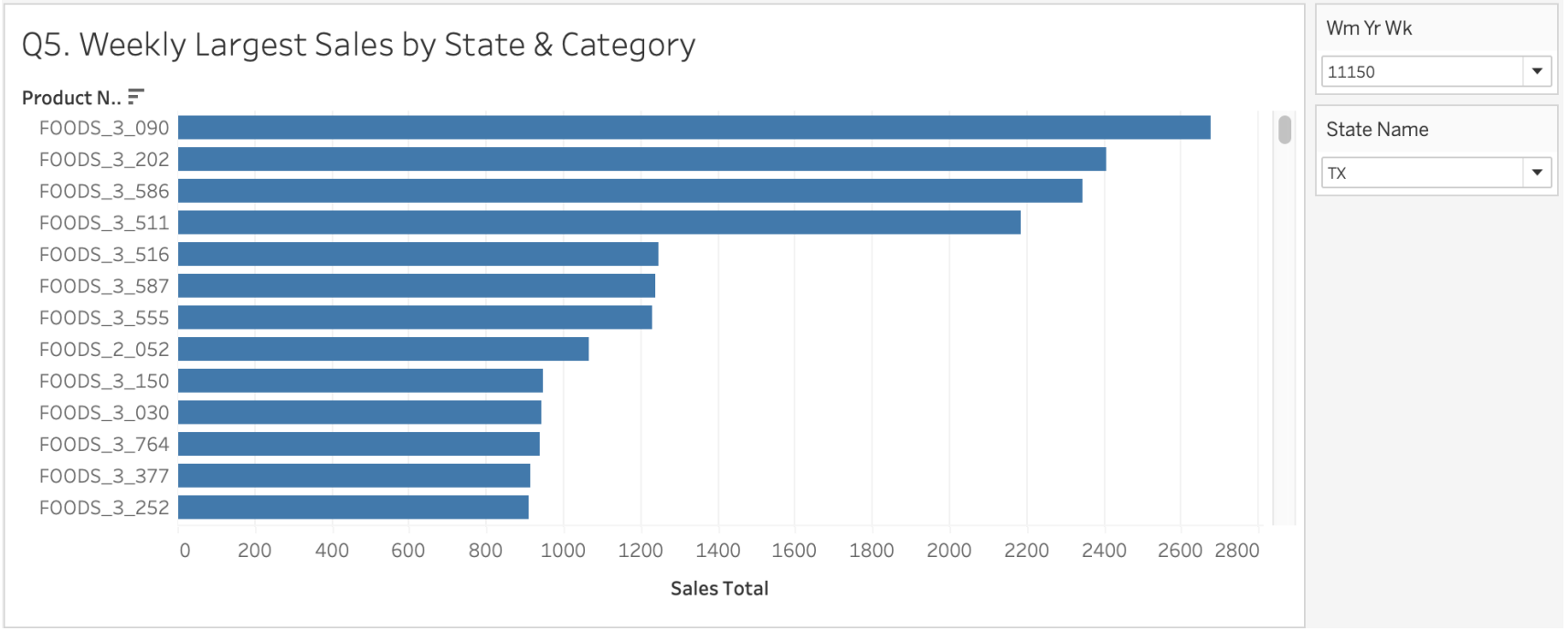


The product FOODS_3_090 had the largest monthly sales quantity in CA in January, roughly 3 times larger than other products.

BOSTON UNIVERSITY

Data Visualization in Tableau

Q5. Which product has the largest weekly sales in Texas in a selected week?



The product FOODS_3_090 had the largest weekly sales in TX in week 11150, slightly larger than the 2nd product.

BOSTON UNIVERSITY

Recap & Improvement

- Created a data warehouse with Walmart sales data.
- Designed normalized and dimensional schema
- Used Python and SQL for ETL processes
- Visualized in Tableau to answer business questions
- Time granularity is important for schema design
- If there was more time, I would
  - create a month fact table
  - create a stored procedure to populate the TYPE2 SCD PRICE dimension
  - add profit data to see which product or category has the potential to increase profit margins
  - add weather data to see how the weather affects the sales.

What I learned

Describe data files

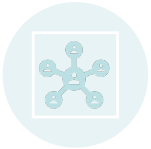
Import flat files into PostgreSQL

Design normalized ERD

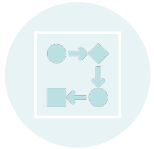
Write queries to move data from flat files to normalized data tables

Design data warehouse schema

ETL data from normalized schema into the warehouse schema



Connect to Tableau



Visualize with Tableau

References

- Dataset: https://www.kaggle.com/competitions/m5-forecasting-accuracy/data
- Sqlalchemy: https://www.geeksforgeeks.org/how-to-insert-a-pandas-dataframe-to-an-existing-postgresql-table/
- Dblink: https://www.postgresql.org/docs/current/contrib-dblink-function.html
- Tableau – PostgreSQL connection: https://www.cdata.com/kb/tech/postgresql-tableau-desktop.rst http://www.dmod-blg.com/dataanalysis/tableau-desktop/datasource/postgresql/
- Tableau: https://help.tableau.com/current/guides/get-started-tutorial/en-us/get-started-tutorial-home.htm
- Datacamp: https://app.datacamp.com/learn/skill-tracks/tableau-fundamentals

Thank you!
Any question?