SANTA CLARA UNIVERSITY

THE JESUIT UNIVERSITY IN THE SILICON VALLEY

Project Presentation
Boba Tea Shop

Project Members - Gaurvi Lal & Steven Wang

Agenda

- Business Description
- Data Collection Details
- Performing Data Analysis
- Normalization
- SQL
  - Views
  - Select Queries
  - Joins & Subqueries
  - Trigger
  - Procedure

Business Description

Shop Description

- Sells speciality flavored teas located close to a big University campus
- Contains about 15 drink options and 4 snack options
- Drinks include
  - Milk Tea
  - Strawberry Tea
  - Thai Tea
  - Lemon Tea
  - Passion Fruit Tea etc.
- Snack Options - Fries, Calamari, Chicken bites and Tofu
- Customers can customize their drinks
  - Drink size
  - Ice level - Light, Medium, High
  - Sugar amount - 25% sweet, 50%, 75% etc.
  - Toppings - Additional Pearls, Taro etc.

Data Collection Details

Below is the summary of data that will be collected -

- Types of drinks ordered - helps in analyzing which are the most popular drinks
- Number of Orders per customer
- Number of people order snacks vs drinks
- How long people spend in the shop
- How much people tip
- Demographics of customers - University students mostly
- Customer satisfaction rating - 1 to 5 stars, or no response

Performing Data Analysis

Questions that can be analyzed from the data

- Most favourable time for the customers
  - This data can help in increasing the staffing at those times
  - Will allow people to have a better experience
  - Eventually leading to increased profits
- What drinks are popular? What are unpopular?
  - Unpopular drinks can be removed from the menu
  - Helps in reducing the costs of buying those supplies/ingredients
  - For popular drinks, we can have promotions to further increase revenue
- Relationship/trend between how much customers tip depending on ratings/demographics/order type/time spent, etc.
- To increase the tip amount
  - How to improve customer experience
  - How about customer ratings

- Improve the store's image as higher ratings could attract more customers

Sample Data

Customer	Gender	Age	Order ID	ServerName	Order(Product ID, ProductName, ProductQty)	Price ($)	TimeSpent	Tip	Rating
A	Male	20	123	1	(1,Milk Boba Tea,1)	7	10	0.8	4
B	Female	18	456	2	(2,Strawberry Boba Tea,2)	10	12	2	4.5
C	Male	19	789	3	(3,Honey Boba Tea,1)	8	15	1.5	4
D	Female	22	221	4	(4,Fries,2)	4	8	0.5	4
E	Male	24	121	5	(5,Chicken Bites,1)	4	6	0.5	1
F	Female	25	432	1	(6,Calamari,1)	4	20	0.5	3
G	Male	30	234	2	(7,Tofu Bites,2)	4	30	0.5	3.5
H	Female	32	568	3	(8,Coffee Milk Tea,2)	7	4	2	5
I	Male	27	908	4	(9,Thai Tea,3)	6	7	0.5	2.5
J	Female	23	678	5	(10,Mango Tea,1)	8	11	1	4

Normalization

1st Denormalized Table

```
CREATE TABLE MasterDenorm
(
   CustomerName  VARCHAR(20) NOT NULL,
Gender VARCHAR(20),
Age INT,
OrderID INT NOT NULL  PRIMARY KEY,
ServerName VARCHAR(20),
Orders VARCHAR(20),
OrderCategory VARCHAR(20), ……
);
```

#Master Table 1NF

```
CREATE TABLE MasterNorm
(
  CustomerName VARCHAR(20) NOT NULL,
  Gender VARCHAR(20),
  Age INT,
  OrderID INT NOT NULL PRIMARY KEY,
  ServerName VARCHAR(20),
  ProductID INT, …….
);
```

#Order Table 2NF

Table 2NF

```
CREATE TABLE Orders
(
  OrderID INT NOT NULL PRIMARY KEY,
  CustomerName VARCHAR(20)
);
```

#Product Table 2NF

```
CREATE TABLE Product
(
  ProductID INT NOT NULL PRIMARY KEY,
  ProductName VARCHAR(50)
);
```

#OrderDetails

```
CREATE TABLE OrderDetails
(
  OrderID INT NOT NULL PRIMARY KEY,
  ProductID INT NOT NULL,
  ProductQuantity INT
);
```

SQL Trigger

Trigger to update a table when a new record is inserted

```
DELIMITER //
CREATE TRIGGER ProductInsertTrigger AFTER INSERT ON OrderDescr
FOR EACH ROW
BEGIN
IF
((SELECT COUNT(*) FROM OrderFrequency WHERE Product = NEW.ProductName) = 0)
THENrderFrequency SELECT ProductName, COUNT(ProductName) FROM OrderDescr
WHERE ProductName = N
INSERT INTO OEW.ProductName GROUP BY ProductName;
ELSE
UPDATE OrderFrequency SET NumOrders = (SELECT COUNT(ProductName) FROM OrderDescr
WHERE ProductName = NEW.ProductName GROUP BY ProductName)
WHERE Product=New.ProductName;
END IF;
END //
DELIMITER ;
```

SQL Views

SQL View to list the products sold to analyze how the items are priced

```
CREATE VIEW PriceCategory AS
SELECT ProductID, ProductName, DefaultPrice,
CASE
        WHEN DefaultPrice > 7 THEN 'High'
        WHEN DefaultPrice > 3 THEN 'Medium'
        WHEN DefaultPrice > 0 THEN 'Low'
        ELSE 'N/A'
END AS PriceRange
FROM OrderDescr;

SELECT * FROM PriceCategory;
```

	ProductID	ProductName	DefaultPrice	PriceRange

SQL Views (contd.)

SQL View to display #hours employees worked - Double OT (Over time), OT, Regular or Reduced Shift

- Helps in redistributing the hours (if necessary)
- Used as a reference when an hour change request is made

```
CREATE VIEW EmployeeWorked AS
SELECT ServerID, ServerName, PayRate, Hours,
CASE
        WHEN Hours > 10 THEN 'Double OT'
        WHEN Hours > 6 THEN 'OT'
        WHEN 4 > 0 THEN 'Regular'
        ELSE 'Reduced Shift'
END AS Worktype
FROM Servers;

SELECT * FROM EmployeeWorked;
```

	ServerID	ServerName	PayRate	Hours	Worktype

SQL Select Queries

Query to determine the popular drink/snack and the #of items ordered

SELECT ProductID, ProductName, COUNT(ProductID) FROM Orders ORDER BY COUNT(ProductID) DESC;

Query to find the most favoured order as determined by

- the Customer Rating and
- the tip amount received

SELECT ProductID, Tip, ReviewRating FROM Orders ORDER BY ReviewRating DESC;

Query to determine the most expensive Order category and their mean prince

SELECT OrderCategory, AVG(DefaultPrice) AS AveragePrice FROM OrderDescr

GROUP BY OrderCategory

ORDER BY AveragePrice DESC;

Query to determine the age group that spent the most time in the store

SELECT Age, AVG(TimeSpent) AS AverageTimeSpent FROM Orders WHERE Age IN(18, 19, 20, 21, 22, 23, 24)
GROUP BY Age ORDER BY AverageTimeSpent DESC;

SQL Joins & Subqueries

To determine the Server, Pay Rate and Shift ID who took the most #Orders sorted by total orders. Helps to determine

- Who works the most #hours and how they are compensated?

```
SELECT Servers.ServerID, Servers.PayRate, Servers.ShiftID,
COUNT(Orders.OrderID) AS OrdersTotal
FROM Servers
INNER JOIN Orders
ON Servers.ServerName = Orders.ServerName GROUP BY  Servers.ServerName
Order BY OrdersTotal DESC;
```

	ServerID	PayRate	ShiftID	OrdersTotal

Usually more Women visit the Boba shops than men. Subquery to determine what is the favoured food / drink choice of women

```
SELECT COUNT(OrderDescr.ProductID) AS ItemCount, OrderDescr.ProductName, OrderDescr.OrderCategory, OrderDescr.DefaultPrice
FROM OrderDescr WHERE
OrderDescr.ProductID IN
        (SELECT Orders.ProductID FROM Orders
   WHERE Gender = 'Female')
GROUP BY OrderDescr.ProductName;
```

	ItemCount	ProductName	OrderCategory	DefaultPrice

SQL Procedure

Compare the customer rating for given two products

```
DELIMITER $$
CREATE PROCEDURE CompareCustomerRating (IN ProdID1 INT, IN ProdID2 INT, OUT CustRatingDiff INT )
BEGIN
    DECLARE CustRating1 INT;
    DECLARE CustRating2 INT;
    SELECT CustomerExperienceRating INTO CustRating1 FROM Orders where Orders.ProductID = ProdID1;
    SELECT CustomerExperienceRating INTO CustRating2 FROM Orders where Orders.ProductID = ProdID2;
            SET CustRatingDiff = CustRating1 - CustRating2;
END $$
DELIMITER ;

CALL CompareCustomerRating(100, 101, @df);
SELECT @df;
```

```
CALL CompareCustomerRating(100, 101, @df);
SELECT @df;
```

	@df
▶	-1

Thank You