Group 2 – Hugo Gaspar, Hudson Peden, Daniel Regan, Peyton Woods

Project 5

10/18/16

“Chapter 3 SQL Statements”

1
Retrieve products that fit within a college student’s budget.

```
SELECT foodid, price
FROM product
WHERE price <= 6
ORDER BY price;
```

2
Retrieve product line items of orders to determine how many unique products were bought on each order.

```
SELECT orderid, foodsequence
FROM orderlineitems
ORDER BY orderid;
```

3
Retrieve all information from orders.

```
SELECT *
FROM orders;
```

4
Retrieve the number of products bought by individual order in which the viewer assumes count refers to the number of products bought.

```
SELECT foodid, foodcount AS count
FROM orderlineitems
ORDER BY foodid;
```

5
Retrieve customers’ full name in order to carry out a new initiative this store is doing by calling customers by their full name.

```
SELECT orderid, orderdate, custfname + ' ' + custlname
FROM orders
ORDER BY orderid;
```

6

Retrieve order totals to determine the average amount per order.

```
SELECT orderid, foodcount*price AS ordertotal
FROM product join orderlineitems
       ON product.foodid=orderlineitems.foodid
ORDER BY orderid;
```

7

A certain Chick-fil-a decided to make the orders more efficient by using customers first and last initials.

```
Select CustFName, CustLName,
    LEFT(CustFName, 1)+
       LEFT(CustLName, 1) AS Initials
From Orders;
```

8

Retrieves how many of each unique product was bought.

```
Select Distinct FoodID,SUM(FoodCount)
from OrderLineItems
group by FoodID
order by FoodID;
```

9

Retrieves the top five highest priced items.

```
Select TOP 5 Price, FoodID
From Product
Order By Price DESC;
```

10

Retrieves all items with prices under $6.00.

```
Select FoodID, Price
From Product
Where Price<=6;
```

11

Retrieves the total order amounts which are greater than $10.00 and were placed from October 1, 2016 and onward.

```
Select price*foodcount as ordertotal
from product join orderlineitems
on product.foodid=orderlineitems.foodid
join orders on orderlineitems.orderid=orders.orderid
where price*foodcount>10 AND orderdate>='10-01-2016';
```

12 IN OPERATOR:

Select all orders in line that have more than 2 of the same type of product.

13 BETWEEN OPERATOR:

Select all products that have price between 4 and 6 in order to help a student decide what he is going to get for dinner.

```
Select *
From Product
Where Price Between 4 AND 6;
```

14 LIKE OPERATOR:

Select all orders that have a customer with the last name 'Li' to find who order it.

```
Select *
From Orders
Where CustLName Like 'Li';
```

15 IS NULL OPERATOR:

Select all orders which customer do not provide their last name.

```
Select *
From Orders
Where CustLName IS NULL;
```

16 SORT A RESULT SET BY A COLUMN NAME:

Select all orders and order them by the last requested.

```
Select *
From Orders
Order by OrderDate DESC;
```

17 SORT A RESULT SET BY AN EXPRESSION:

Select all orders and order them by the first and last name of the customers.

```
Select *
From Orders
Order by CustFName + CustLName;
```

18 RETRIEVE A RANGE OF SELECTED ROWS:

Select all products and order them from the smallest to highest price, and presenting only the 5 lowest prices.

```
Select *
From Product
Order by Price ASC
       OFFSET 0 Rows
       FETCH First 5 Rows Only;
```