**Penjelasan Query Nomor 1**

Query tersebut digunakan untuk menghasilkan laporan mengenai pembelian makanan yang dilakukan oleh staff pria (*male*) pada tahun 2019. Laporan ini terdiri dari StaffName dan PurchaseDate serta jumlah total makanan yang dibeli oleh setiap staff pada setiap tanggal pembelian. Berikut adalah penjelasan dari query tersebut:

1. SELECT UPPER(StaffName) AS 'Staff Name', PurchaseDate, SUM(pdf.FoodQtyPurchase) as 'Total Food Purchase' FROM MsStaff ms

* UPPER(StaffName) AS 'Staff Name': berfungsi dalam menampilkan nama staff dalam huruf kapital dengan nama kolom 'Staff Name'.
* PurchaseDate: kolom ini akan menampilkan tanggal pembelian.
* SUM(pdf.FoodQtyPurchase) as 'Total Food Purchase: berfungsi dalam menghitung jumlah total makanan yang dibeli dan menampilkannya dengan nama kolom 'Total Food Purchase'.
* FROM MsStaff ms: menandakan bahwa tabel yang digunakan untuk mendapatkan data yang diinginkan berasal dari tabel MsStaff

1. JOIN PurchaseHeader ph ON ms.StaffID = ph.StaffID:

* Bagian ini menghubungkan tabel MsStaff dengan tabel PurchaseHeader berdasarkan StaffID yang sama.

1. JOIN PurchaseDetailFood pdf ON ph.PurchaseID = pdf.PurchaseID:

* Bagian ini menghubungkan tabel PurchaseHeader dengan tabel PurchaseDetailFood berdasarkan PurchaseID yang sama.

1. WHERE StaffGender = 'Male' AND YEAR(PurchaseDate) = 2019:

* Bagian ini berfungsi dalam memilih data dari staff yang berjenis kelamin pria (StaffGender = 'Male') dan yang melakukan pembelian pada tahun 2019 (YEAR(PurchaseDate) = 2019).

1. GROUP BY StaffName, PurchaseDate:

* Bagian ini mengelompokkan hasil query berdasarkan nama staff dan tanggal pembelian.

Tujuan dari query tersebut adalah untuk menganalisis dan melaporkan jumlah total makanan yang dibeli oleh staff pria (*male*) selama tahun 2019. Laporan ini memberikan informasi rinci tentang siapa yang membeli, kapan pembelian dilakukan, dan berapa banyak makanan yang dibeli.

A screen shot of a document

Description automatically generated

**Penjelasan Query Nomor 2**

Query tersebut digunakan untuk menghasilkan laporan mengenai transaksi pembelian minuman yang dilakukan oleh supplier, dengan syarat jumlah pembelian minuman kurang dari 5 dan ID pembelian memiliki nomor urut yang genap. Laporan ini mencakup PurchaseID, SupplierName dalam huruf kecil, dan jumlah total minuman yang dibeli. Berikut adalah penjelasan dari query tersebut:

1. SELECT PH.PurchaseID, LOWER(ms.SupplierName) AS 'Supplier Name', SUM(PurchaseDetailDrink.DrinkQtyPurchase) AS [Total Drink Purchase] FROM PurchaseHeader PH:

* PH.PurchaseID: berfungsi dalam menampilkan ID pembelian.
* LOWER(ms.SupplierName) AS 'Supplier Name': SupplierName akan ditampilkan dalam huruf kecil dengan nama kolom 'Supplier Name'.
* SUM(PurchaseDetailDrink.DrinkQtyPurchase) AS [Total Drink Purchase]: berfungsi dalam menghitung jumlah total minuman yang dibeli dan menampilkannya dengan nama kolom 'Total Drink Purchase'.
* FROM PurchaseHeader PH: menandakan bahwa tabel yang digunakan untuk mendapatkan data yang diinginkan berasal dari tabel PurchaseHeader.

1. JOIN MsSupplier ms ON PH.SupplierID = ms.SupplierID:

* Bagian ini menghubungkan tabel PurchaseHeader dengan tabel MsSupplier berdasarkan SupplierID yang sama.

1. JOIN PurchaseDetailDrink ON PurchaseDetailDrink.PurchaseID = PH.PurchaseID:

* Bagian ini menghubungkan tabel PurchaseHeader dengan tabel PurchaseDetailDrink berdasarkan PurchaseID yang sama.

1. WHERE PurchaseDetailDrink.DrinkQtyPurchase < 5:

* Berfungsi dalam memilih transaksi di mana jumlah pembelian minuman kurang dari 5 unit.

1. AND CONVERT(INT, SUBSTRING(PH.PurchaseID, 3, LEN(PH.PurchaseID))) % 2 = 0:

* Berfungsi dalam memilih transaksi di mana bagian angka dari PurchaseID adalah genap.

1. GROUP BY PH.PurchaseID, ms.SupplierName:

* Bagian ini mengelompokkan hasil query berdasarkan nama PurchaseID dan SupplierName.

Tujuan dari query tersebut adalah untuk mendapatkan gambaran mengenai transaksi pembelian minuman dari supplier, yang memenuhi kriteria tertentu, yaitu jumlah pembelian kurang dari 5 dan memiliki PurchaseID dengan nomor urut genap. Laporan ini memberikan informasi rinci tentang transaksi tertentu yang dilakukan oleh Supplier, termasuk PurchaseID, SupplierName, dan total jumlah minuman yang dibeli.

**A screenshot of a phone

Description automatically generated**

**Penjelasan Query Nomor 3**

Query tersebut digunakan untuk menghasilkan laporan mengenai harga tertinggi makanan dan harga terendah minuman pada setiap tanggal transaksi sebelum bulan Juni tahun 2023. Laporan ini mencakup tanggal transaksi, harga makanan tertinggi, dan harga minuman terendah. Berikut adalah penjelasan dari query tersebut:

1. SELECT CONVERT(VARCHAR, TransactionDate, 107) AS Transaction\_Date, MAX(Food.FoodPrice) AS Highest\_Food\_Price, MIN(Drink.DrinkPrice) AS Lowest\_Drink\_Price FROM TransactionHeader:

* CONVERT(VARCHAR, TransactionDate, 107) AS Transaction\_Date: mengonversi tanggal transaksi menjadi format teks dengan gaya 'Mon DD, YYYY' dan memberikan nama kolom 'Transaction\_Date'.
* MAX(Food.FoodPrice) AS Highest\_Food\_Price: berfungsi dalam menghitung harga tertinggi dari makanan yang dibeli pada setiap tanggal transaksi dan memberikan nama kolom 'Highest\_Food\_Price'.
* MIN(Drink.DrinkPrice) AS Lowest\_Drink\_Price: berfungsi dalam harga terendah dari minuman yang dibeli pada setiap tanggal transaksi dan memberikan nama kolom 'Lowest\_Drink\_Price'.
* FROM TransactionHeader: menandakan bahwa tabel yang digunakan untuk mendapatkan data yang diinginkan berasal dari tabel TransactionHeader.

1. JOIN TransactionDetail ON TransactionDetail.TransactionID = TransactionHeader.TransactionID:

* Bagian ini menghubungkan tabel TransactionHeader dengan tabel TransactionDetail berdasarkan TransactionID yang sama.

1. JOIN MsFood AS Food ON Food.FoodID = TransactionDetail.FoodID:

* Bagian ini menghubungkan tabel TransactionDetail dengan tabel MsFood berdasarkan FoodID yang sama.

1. JOIN MsDrink AS Drink ON Drink.DrinkID = TransactionDetail.DrinkID:

* Bagian ini menghubungkan tabel TransactionDetail dengan tabel MsDrink berdasarkan DrinkID yang sama.

1. WHERE MONTH(TransactionDate) < 6 AND YEAR(TransactionDate) < 2023:

* Berfungsi dalam memilih transaksi yang terjadi sebelum bulan Juni dan sebelum tahun 2023.

1. GROUP BY TransactionDate:

* Bagian ini mengelompokkan hasil query berdasarkan tanggal transaksi dan SupplierName.

Tujuan dari query tersebut adalah untuk mendapatkan informasi mengenai harga makanan dan minuman pada transaksi yang terjadi sebelum bulan Juni 2023 (06-2023). Laporan ini memberikan informasi rinci tentang setiap TransactionDate, termasuk harga makanan tertinggi dan harga minuman terendah pada hari tersebut.

A screenshot of a phone

Description automatically generated

**Penjelasan Query Nomor 4**

Query tersebut digunakan untuk menghasilkan laporan mengenai pembelian makanan kategori Fried oleh staff, dengan syarat jumlah pembelian lebih dari 2. Laporan ini mencakup StaffName, FoodCategory, rata-rata (Average) pembelian makanan, dan total (Sum) pembelian makanan. Berikut adalah penjelasan dari query tersebut:

1. SELECT LOWER(SUBSTRING(MsStaff.StaffName, 1, CHARINDEX(' ', MsStaff.StaffName) - 1)) AS 'Name', FoodCategory, AVG(PurchaseDetailFood.FoodQtyPurchase) AS 'Average Total Food Purchase', SUM(PurchaseDetailFood.FoodQtyPurchase) AS [Total Food Purchase] FROM MsStaff:

* LOWER(SUBSTRING(MsStaff.StaffName, 1, CHARINDEX(' ', MsStaff.StaffName) - 1)): memperoleh substring dari nama staff hingga karakter sebelum spasi pertama, lalu mengubah semuanya menjadi huruf kecil.
* FoodCategory: menampilkan kategori makanan
* AVG(PurchaseDetailFood.FoodQtyPurchase) AS 'Average Total Food Purchase': menghitung rata-rata jumlah makanan yang dibeli dengan nama kolom 'Average Total Food Purchase'
* SUM(PurchaseDetailFood.FoodQtyPurchase) AS [Total Food Purchase]: menghitung total jumlah makanan yang dibeli dengan nama kolom 'Total Food Purchase'
* FROM MsStaff : menandakan bahwa tabel yang digunakan untuk mendapatkan data yang diinginkan berasal dari tabel TransactionHeader.

1. JOIN PurchaseHeader ON PurchaseHeader.StaffID = MsStaff.StaffID

* Bagian ini menghubungkan tabel MsStaff dengan tabel PurchaseHeader berdasarkan StaffID yang sama.

1. JOIN PurchaseDetailFood ON PurchaseDetailFood.PurchaseID = PurchaseHeader.PurchaseID

* Bagian ini menghubungkan tabel PurchaseHeader dengan tabel PurchaseDetailFood berdasarkan PurchaseID yang sama.

1. JOIN MsFood ON MsFood.FoodID = PurchaseDetailFood.FoodID

* Bagian ini menghubungkan tabel PurchaseDetailFood dengan tabel MsFood berdasarkan FoodID yang sama.

1. WHERE PurchaseDetailFood.FoodQtyPurchase > 2 AND MsFood.FoodCategory = 'Fried'

* Befungsi dalam memilih transaksi di mana jumlah pembelian makanan lebih dari 2 dan kategori makanan adalah Fried.

1. GROUP BY StaffName, FoodCategory

* Bagian ini mengelompokkan hasil query berdasarkan StaffName dan FoodCategory.

Tujuan dari query tersebut adalah untuk mendapatkan informasi mengenai pembelian makanan kategori Fried oleh staff dengan pembelian lebih dari 2. Laporan ini memberikan informasi rinci tentang siapa yang membeli, kategori makanan apa yang dibeli, rata-rata jumlah pembelian, dan total jumlah pembelian.

A screenshot of a graph

Description automatically generated

**Penjelasan Query Nomor 5**

Query tersebut digunakan untuk mendapatkan data transaksi minuman yang telah dibeli untuk tahun berikutnya untuk kategori minuman Herbal dan Soft Drink. Laporan ini terdiri dari TransactionID, Tanggal Transaksi yang ditambah tahun nya, dan Drink Quantity, yang menunjukkan tanggal perkiraan transaksi minuman pada tahun berikutnya serta jumlah minuman yang terjual dalam satuan Cup. Berikut adalah penjelasan dari query tersebut:

1. SELECT DISTINCT th.TransactionID, DATEADD(YEAR, 1, TransactionDate) AS [Drink Transaction Forecast], CONCAT(DrinkQtySales, ' Cup') AS [Drink Quantity] FROM TransactionDetail td:

* SELECT DISTINCT th.TransactionID: Memilih TransactionID dari tabel TransactionHeader tanpa adanya redundancy.
* DATEADD(YEAR, 1, TransactionDate) AS [Drink Transaction Forecast]: menggunakan fungsi DATEADD untuk menambahkan satu tahun ke tanggal transaksi (TransactionDate) dan memberikan hasilnya sebagai nama kolom 'Drink Transaction Forecast'.
* CONCAT(DrinkQtySales, ' Cup') AS [Drink Quantity]: menggabungkan jumlah penjualan minuman (DrinkQtySales) dengan string ' Cup', dan memberikan nama kolom sebagai 'Drink Quantity'.
* FROM TransactionDetail td: menandakan bahwa tabel utama yang digunakan untuk mendapatkan data adalah tabel TransactionDetail.

1. JOIN TransactionHeader th ON td.TransactionID = th.TransactionID

* Bagian ini menghubungkan tabel TransactionDetail dengan tabel TransactionHeader berdasarkan TransactionID yang sama.

1. (SELECT DrinkID FROM MsDrink WHERE DrinkCategory IN ('Herbal', 'Soft Drink')) AS CategoryDrink

* Subquery ini memilih DrinkID dari tabel MsDrink dimana DrinkCategory adalah 'Herbal' atau 'Soft Drink'. Subquery ini diberi alias CategoryDrink.

1. WHERE td.DrinkID = CategoryDrink.DrinkID

* Memfilter hasil agar hanya menyertakan transaksi dimana DrinkID di tabel TransactionDetail sesuai dengan DrinkID dari subquery CategoryDrink.

Tujuan dari query ini adalah untuk mendapat data dari minuman kategori Herbal dan Soft Drink dengan menambahkan satu tahun ke tanggal transaksi dan menyajikan jumlah minuman yang terjual dalam satuan 'Cup'.

A screenshot of a table

Description automatically generated

**Penjelasan Query Nomor 6**

Query tersebut digunakan untuk menghasilkan laporan mengenai transaksi film yang dilakukan oleh staff dengan kategori film 'MO003' yang memiliki durasi di atas rata-rata. Laporan ini terdiri dari StaffID, Transaction Date, Movie Identification, Movie Name, dan Movie Category. Berikut adalah penjelasan dari query tersebut:

1. SELECT ms.StaffID, CONVERT(VARCHAR, TransactionDate, 106) AS [Transaction Date], REPLACE(mm.MovieID, 'MO', 'Movie ') AS [Movie Identification], CONCAT('Film', MovieName) AS [Movie Name], MovieCategory FROM MsStaff ms\*\*:

* ms.StaffID: menampilkan StaffID yang melakukan transaksi.
* CONVERT(VARCHAR, TransactionDate, 106) AS [Transaction Date]: mengubah format tanggal transaksi menjadi string dengan format 'DD Mon YYYY' dan memberikan nama kolom sebagai 'Transaction Date'.
* REPLACE(mm.MovieID, 'MO', 'Movie ') AS [Movie Identification]: menggantikan 'MO' dalam MovieID dengan 'Movie ' dan memberikan nama kolom sebagai 'Movie Identification'.
* CONCAT('Film', MovieName) AS [Movie Name]: menggabungkan kata 'Film' dengan MovieName dan memberikan nama kolom sebagai 'Movie Name'.
* MovieCategory: menampilkan kategori film.
* FROM MsStaff ms: enandakan bahwa tabel utama yang digunakan untuk mendapatkan data yang diinginkan berasal dari tabel MsStaff.

1. JOIN TransactionHeader th ON ms.StaffID = th.StaffID

* Bagian ini menghubungkan tabel MsStaff dengan tabel TransactionHeader berdasarkan StaffID yang sama.

1. JOIN TransactionDetail td ON th.TransactionID = td.TransactionID

* Bagian ini menghubungkan tabel TransactionHeader dengan tabel TransactionDetail berdasarkan TransactionID yang sama.

1. JOIN MsMovie mm ON td.MovieID = mm.MovieID

* Bagian ini menghubungkan tabel TransactionDetail dengan tabel MsMovie berdasarkan MovieID yang sama.

1. (SELECT AVG(MovieDuration) AS Average FROM MsMovie) AS AverageMovieDurationSubque

* menghitung rata-rata durasi film dari tabel MsMovie dan memberikan hasilnya sebagai nama kolom 'AverageMovieDuration'.

1. WHERE mm.MovieID = 'MO003' AND MovieDuration > Average

* Bagian ini berfungsi dalam memilih data dari tabel MsMovie dengan MovieID 'MO003' dan durasi film (MovieDuration) yang lebih besar dari rata-rata durasi film.

Tujuan dari query ini adalah untuk menghasilkan laporan mengenai transaksi film yang dilakukan oleh staff yang menonton film dengan kategori 'MO003' dan memiliki durasi di atas rata-rata. Laporan ini memberikan informasi rinci tentang siapa yang menonton film, kapan transaksi dilakukan, identifikasi film, nama film, dan kategori film.

A screen shot of a movie

Description automatically generated

**Penjelasan Query Nomor 7**

Query tersebut digunakan untuk menghasilkan laporan mengenai penjualan tiket film yang dilakukan oleh pelanggan pada kuartal pertama tahun tertentu. Laporan ini terdiri dari Last Name dan Total Movie Sold, yang menunjukkan nama belakang pelanggan dan total penjualan tiket film yang melebihi rata-rata penjualan tiket. Berikut adalah penjelasan dari query tersebut:

1. SELECT SUBSTRING(CustomerName, CHARINDEX(' ', CustomerName), LEN(CustomerName)) AS [Last Name], CAST(SUM(TicketQtySales\*MoviePrice) AS NUMERIC(10,2)) AS [Total Movie Sold] FROM MsCustomer mc:

* SUBSTRING(CustomerName, CHARINDEX(' ', CustomerName), LEN(CustomerName)) AS [Last Name]\*\*: mengambil bagian nama belakang dari CustomerName dan memberikan nama kolom sebagai 'Last Name'.
* CAST(SUM(TicketQtySales\*MoviePrice) AS NUMERIC(10,2)) AS [Total Movie Sold]: menghitung total penjualan tiket film (kuantitas tiket dikalikan dengan harga film), kemudian mengkonversinya menjadi tipe numerik dengan dua desimal dan memberikan nama kolom sebagai 'Total Movie Sold'.
* FROM MsCustomer mc: menandakan bahwa tabel utama yang digunakan untuk mendapatkan data yang diinginkan berasal dari tabel MsCustomer.

1. JOIN TransactionHeader th ON mc.CustomerID = th.CustomerID

* Bagian ini menghubungkan tabel MsCustomer dengan tabel TransactionHeader berdasarkan CustomerID yang sama.

1. JOIN TransactionDetail td ON th.TransactionID = td.TransactionID

* Bagian ini menghubungkan tabel TransactionHeader dengan tabel TransactionDetail berdasarkan TransactionID yang sama.

1. JOIN MsMovie mm ON td.MovieID = mm.MovieID

* Bagian ini menghubungkan tabel TransactionDetail dengan tabel MsMovie berdasarkan MovieID yang sama.

1. WHERE DATENAME(QUARTER, TransactionDate) = 1

* Bagian ini berfungsi untuk memilih data dari transaksi yang dilakukan pada kuartal pertama tahun tersebut.

1. GROUP BY CustomerName

* Bagian ini mengelompokkan hasil query berdasarkan CustomerName.

1. HAVING SUM(TicketQtySales \* MoviePrice) > (SELECT AVG(TotalMovieSold) FROM(...))

* Menyaring hasil untuk hanya menyertakan kelompok dengan total penjualan tiket yang lebih besar dari rata-rata penjualan tiket.
* Subquery di dalam HAVING menghitung rata-rata penjualan tiket (dalam bentuk TotalMovieSold) untuk semua transaksi.

Tujuan dari query ini adalah untuk menghasilkan laporan mengenai penjualan tiket film oleh pelanggan yang melebihi rata-rata penjualan tiket pada kuartal pertama tahun tertentu. Laporan ini memberikan informasi rinci tentang nama belakang CustomerName dan total penjualan tiket film yang telah dilakukan.



**Penjelasan Query Nomor 8**

Query tersebut digunakan untuk menghasilkan laporan mengenai transaksi yang dilakukan oleh pelanggan wanita *(female).* Laporan ini terdiri dari TransactionID, CustomerName, dan TransactionDate, serta menunjukkan transaksi yang dilakukan oleh setiap pelanggan yang jumlah transaksinya melebihi rata-rata. Berikut adalah penjelasan dari query tersebut:

1. SELECT UPPER(REPLACE(th.TransactionID, 'TR', 'Transaction ')) AS [Transaction], StaffName, ('Mr/Mrs. ' + CustomerName) AS [Customer Name], YEAR(TransactionDate) AS [Transaction Date] FROM MsCustomer mc

* UPPER(REPLACE(th.TransactionID, 'TR', 'Transaction ')) AS [Transaction]\*\*: Mengganti 'TR' pada TransactionID dengan 'Transaction' dan mengubahnya menjadi huruf kapital, memberikan alias sebagai 'Transaction'.
* StaffName: menampilkan nama staff yang terkait dengan transaksi.
* ('Mr/Mrs. ' + CustomerName) AS [Customer Name]: enambahkan 'Mr/Mrs. ' di depan nama pelanggan dan memberikan nama kolom sebagai 'Customer Name'.
* YEAR(TransactionDate) AS [Transaction Date]: mengambil tahun dari tanggal transaksi dan memberikan nama kolom sebagai 'Transaction Date'.

1. JOIN TransactionHeader th ON th.CustomerID = mc.CustomerID

* Bagian ini menghubungkan tabel MsCustomer dengan tabel TransactionHeader berdasarkan CustomerID yang sama.

1. JOIN TransactionDetail td ON td.TransactionID = th.TransactionID

* Bagian ini menghubungkan tabel TransactionHeader dengan tabel TransactionDetail berdasarkan TransactionID yang sama.

1. JOIN MsStaff ON MsStaff.StaffID = th.StaffID

* Bagian ini menghubungkan tabel TransactionHeader dengan table MsStaff berdasarkan StaffID yang sama.

1. (SELECT AVG(TotalTransaction) AS Average FROM (SELECT COUNT(th.TransactionID) AS TotalTransaction FROM TransactionHeader th JOIN TransactionDetail td ON th.TransactionID = td.TransactionID GROUP BY th.TransactionID) AS Total) AS AverageTransaction

* Subquery ini menghitung rata-rata jumlah transaksi.
* Bagian dalam subquery pertama menghitung total transaksi per TransactionID.
* Bagian luar subquery menghitung rata-rata dari total transaksi tersebut dan memberikan nama kolom sebagai Average.

1. WHERE mc.CustomerGender = 'Female'

* Memfilter data untuk hanya menyertakan pelanggan wanita.

1. GROUP BY Average, th.TransactionID, CustomerName, TransactionDate, StaffName

* Mengelompokkan hasil query berdasarkan rata-rata transaksi, TransactionID, CustomerName, TransactionDate, dan StaffName.

1. HAVING COUNT(th.TransactionID) > Average

* Menyaring hasil agar hanya menyertakan transaksi dimana jumlah transaksi melebihi rata-rata yang telah dihitung dalam subquery.

Tujuan dari query ini adalah untuk menghasilkan laporan mengenai transaksi yang dilakukan oleh pelanggan wanita yang jumlah transaksinya melebihi rata-rata. Laporan ini memberikan informasi rinci tentang TransactionID, CustomerName, TransactionDate, dan StaffName

A screenshot of a cellphone

Description automatically generated

**Penjelasan Query Nomor 9**

Query tersebut digunakan untuk membuat tampilan (view) bernama "TotalPurchase" yang menyediakan laporan mengenai pembelian tiket film oleh staff yang memiliki rating film 5, di mana jumlah total tiket yang dibeli lebih besar dari rata-rata tiket yang dibeli. Laporan ini mencakup StaffName, MovieName, MovieRating, rata-rata jumlah tiket yang dibeli, dan total tiket yang dibeli. Berikut adalah penjelasan dari query tersebut:

1. CREATE VIEW TotalPurchase AS

* Membuat tampilan (view) baru bernama [TotalPurchase].

1. SELECT REPLACE(StaffName, SUBSTRING(StaffName, 0, CHARINDEX(' ', StaffName)), 'Staff ') AS [Staff], MovieName, MovieRating, AVG(TicketQtySales) AS [Average Ticket Bought], SUM(TicketQtySales) AS [Total Ticket Bought] FROM MsStaff ms

* REPLACE(StaffName, SUBSTRING(StaffName, 0, CHARINDEX(' ', StaffName)), 'Staff '): bertujuan untuk mengganti bagian pertama dari StaffName sebelum spasi pertama dengan 'Staff '
* MovieName: menampilkan nama film
* MovieRating: menampilkan rating film
* AVG(TicketQtySales) AS [Average Ticket Bought]: menghitung rata-rata jumlah tiket yang dibeli dengan nama kolom 'Average Ticket Bought'.
* SUM(TicketQtySales) AS [Total Ticket Bought]: menghitung total jumlah tiket yang dibeli dengan nama kolom 'Total Ticket Bought'.
* FROM MsStaff ms: menandakan bahwa tabel yang digunakan untuk mendapatkan data yang diinginkan berasal dari tabel MsStaff.

1. JOIN TransactionHeader th ON ms.StaffID = th.StaffID

* Bagian ini menghubungkan tabel MsStaff dengan tabel TransactionHeader berdasarkan StaffID yang sama.

1. JOIN TransactionDetail td ON th.TransactionID = td.TransactionID

* Bagian ini menghubungkan tabel TransactionHeader dengan tabel TransactionDetail berdasarkan TransactionID yang sama.

1. JOIN MsMovie mm ON td.MovieID = mm.MovieID

* Bagian ini menghubungkan tabel TransactionDetail dengan tabel MsMovie berdasarkan MovieID yang sama.

1. WHERE MovieRating = 5

* Befungsi dalam memilih transaksi di mana rating film adalah 5.

1. GROUP BY StaffName, MovieName, MovieRating

* Bagian ini mengelompokkan hasil query berdasarkan StaffName, MovieName, dan MovieRating.

1. HAVING SUM(TicketQtySales) > AVG(TicketQtySales)

* Befungsi dalam memilih transaksi di mana total tiket yang dibeli lebih besar dari rata-rata tiket yang dibeli.

Tujuan dari query tersebut adalah untuk mendapatkan data pembelian tiket film oleh staff yang memiliki rating film 5, dengan fokus pada transaksi di mana jumlah total tiket yang dibeli melebihi rata-rata tiket yang dibeli. Laporan ini memberikan informasi rinci tentang siapa yang membeli, MovieName, Movierating, rata-rata pembelian tiket, dan total pembelian tiket.

A screenshot of a computer

Description automatically generated

**Penjelasan Query Nomor 10**

Query tersebut digunakan untuk membuat tampilan (view) bernama "Food Sales" yang menyediakan laporan mengenai penjualan makanan kategori Sandwich dalam tahun berjalan. Laporan ini mencakup FoodName, total jumlah makanan yang terjual, dan harga rata-rata makanan tersebut. Berikut adalah penjelasan dari query tersebut:

1. CREATE VIEW [Food Sales] AS

* Membuat tampilan (view) baru bernama [Food Sales].

1. SELECT FoodName, SUM(FoodQtySales) AS [Total Quantity Sold], ROUND(AVG(FoodPrice), 2) AS 'Average Food Price' FROM MsFood mf:

* FoodName: menampilkan nama makanan
* SUM(FoodQtySales) AS [Total Quantity Sold]: menghitung total jumlah makanan yang terjual dengan nama kolom 'Total Quantity Sold'.
* ROUND(AVG(FoodPrice), 2) AS 'Average Food Price': menghitung rata-rata harga makanan yang terjual dan membulatkannya hingga dua tempat desimal dengan nama kolom 'Average Food Price'.
* FROM MsFood mf: menandakan bahwa tabel yang digunakan untuk mendapatkan data yang diinginkan berasal dari tabel MsFood.

1. JOIN TransactionDetail td ON td.FoodID = mf.FoodID

* Bagian ini menghubungkan tabel MsFood dengan tabel TransactionDetail berdasarkan FoodID yang sama.

1. JOIN TransactionHeader th ON td.TransactionID = th.TransactionID

* Bagian ini menghubungkan tabel TransactionDetail dengan tabel TransactionHeader berdasarkan TransactionID yang sama.

1. WHERE FoodCategory = 'Sandwich' AND YEAR(TransactionDate) = YEAR(GETDATE()):

* Berfungsi dalam memilih transaksi di mana kategori makanan adalah Sandwich dan transaksi terjadi dalam tahun berjalan (tahun yang sama dengan tanggal saat ini).

1. GROUP BY FoodName

* Bagian ini mengelompokkan hasil query berdasarkan FoodName.

Tujuan dari query tersebut adalah untuk memperoleh data penjualan makanan kategori Sandwich selama tahun yang sedang berjalan saat ini. Laporan ini memberikan informasi rinci tentang jenis makanan yang terjual, total kuantitas yang terjual, dan rata-rata harga makanan tersebut.

A screenshot of a computer

Description automatically generated