GROUP PROJECT

OUR TEAM

1. Maitri Vathada
2. Ashly Felix

ABSTRACT

In this project we'll discuss how to use the following SQL clauses using WideWorldImports Database:

1. FROM
2. WHERE
3. GROUP BY
4. HAVING
5. SELECT
6. ORDER BY

PROBLEMS

1. Without SELECT and FROM, we wouldn’t be able to extract data from a database.
2. Without WHERE, we would always retrieve all records, even if we only need a subset of data.
3. Without GROUP BY, HAVING, and ORDER BY, we couldn't summarize, filter, or organize data efficiently.

GOALS

1. Our goal for this project is to learn how to properly use SQL clauses.
2. Our goal also consist of making 10 propositions and be able to execute them and explain them

10 Prepositions

```sql
USE WideWorldImporters
SELECT CustomerID, CustomerName, CustomerCategoryID
FROM Sales.Customers
WHERE CustomerCategoryID = 7;
```

Results | Messages

	CustomerID	CustomerName	CustomerCategoryID
1	801	Eric Torres	7
2	802	Cosmina Vlad	7
3	810	Alena Kellnerova	7
4	813	Shyam Poddar	7
5	831	Bhaavan Rai	7
6	838	Pavel Bogdanov	7
7	843	Marija Justina Pasek	7
8	845	Emilie Hrdlickova	7
9	856	Satish Mittal	7
10	860	Marcela Lucescu	7
11	863	Jakub Lukes	7

```sql
USE WideWorldImporters
SELECT CustomerID, CustomerName, CustomerCategoryID
FROM Sales.Customers
WHERE CustomerCategoryID = 8;
```

Results | Messages

	CustomerID	CustomerName	CustomerCategor…

```sql
USE WideWorldImporters
SELECT CustomerID, CustomerName, CustomerCategoryID
FROM Sales.Customers
WHERE CustomerName = 'Tailspin Toys (Head Office)';
```

Results | Messages

	CustomerID	CustomerName	CustomerCategoryID
1	1	Tailspin Toys (Head Office)	3

```sql
USE WideWorldImporters
SELECT CustomerID, CustomerName, CustomerCategoryID
FROM Sales.Customers
WHERE CustomerName LIKE 'Tailspin Toys%';
```

Results | Messages

	CustomerID	CustomerName	CustomerCategoryID
191	191	Tailspin Toys (Orchard Hill, GA)	3
192	192	Tailspin Toys (Optimo, NM)	3
193	193	Tailspin Toys (Knifley, KY)	3
194	194	Tailspin Toys (Naples Park, FL)	3
195	195	Tailspin Toys (Lesslie, SC)	3
196	196	Tailspin Toys (Howells, NE)	3
197	197	Tailspin Toys (Magalia, CA)	3
198	198	Tailspin Toys (Buell, MO)	3
199	199	Tailspin Toys (Antonito, CO)	3
200	200	Tailspin Toys (Tooele, UT)	3
201	201	Tailspin Toys (Skyway, WA)	3

```sql
USE WideWorldImporters
SELECT CustomerID, CustomerName, CustomerCategoryID
FROM Sales.Customers
WHERE CustomerName LIKE '%Toys%';
```

Results | Messages

	CustomerID	CustomerName	CustomerCategoryID
197	197	Tailspin Toys (Magalia, CA)	3
198	198	Tailspin Toys (Buell, MO)	3
199	199	Tailspin Toys (Antonito, CO)	3
200	200	Tailspin Toys (Tooele, UT)	3
201	201	Tailspin Toys (Skyway, WA)	3
202	401	Wingtip Toys (Head Office)	3
203	402	Wingtip Toys (Black Lick, PA)	3
204	403	Wingtip Toys (Queen Valley, AZ)	3
205	404	Wingtip Toys (Penns Creek, PA)	3
206	405	Wingtip Toys (Bourbonnais, IL)	3
207	406	Wingtip Toys (Tuscaloosa, AL)	3

```sql
USE WideWorldImporters
SELECT CustomerID, SalespersonPersonID
FROM Sales.Orders
WHERE SalespersonPersonID = 3;
```

Results | Messages

	CustomerID	SalespersonPersonID
7271	1018	3
7272	466	3
7273	835	3
7274	119	3
7275	810	3
7276	840	3
7277	838	3
7278	915	3
7279	498	3
7280	1018	3
7281	466	3

```sql
USE WideWorldImporters
SELECT CustomerID, SalespersonPersonID
FROM Sales.Orders
WHERE SalespersonPersonID = 2 and CustomerID = 414;
```

Results | Messages

	CustomerID	SalespersonPersonID
12	414	2
13	414	2
14	414	2
15	414	2
16	414	2
17	414	2
18	414	2
19	414	2
20	414	2
21	414	2
22	414	2

```sql
USE WideWorldImporters
SELECT OrderID, CustomerID, SalespersonPersonID, OrderDate
FROM Sales.Orders
WHERE OrderID IN (1,2,3)
```

Results | Messages

	OrderID	CustomerID	SalespersonPersonID	OrderDate
1	1	832	2	2013-01-01
2	2	803	8	2013-01-01
3	3	105	7	2013-01-01

```sql
USE WideWorldImporters
SELECT OrderID, CustomerID, SalespersonPersonID, OrderDate
FROM Sales.Orders
WHERE OrderID NOT IN (1,2,3);
```

Results | Messages

	OrderID	CustomerID	SalespersonPersonID	OrderDate
1	4	57	16	2013-01-01
2	5	905	3	2013-01-01
3	6	976	13	2013-01-01
4	7	575	8	2013-01-01
5	8	964	7	2013-01-01
6	9	77	7	2013-01-01
7	10	191	20	2013-01-01
8	11	586	3	2013-01-01
9	12	529	15	2013-01-01
10	13	473	13	2013-01-01
11	14	870	8	2013-01-01

```sql
USE WideWorldImporters
SELECT OrderID, CustomerID, SalespersonPersonID, OrderDate
FROM Sales.Orders
WHERE OrderID > 73590;
```

Results | Messages

	OrderID	CustomerID	SalespersonPersonID	OrderDate
1	73591	150	20	2016-05-31
2	73592	1028	13	2016-05-31
3	73593	417	2	2016-05-31
4	73594	552	16	2016-05-31
5	73595	11	14	2016-05-31

- I’m proposing an analysis of sales customer from the from the Sales.Customers table, specifically the phone number, postal city ID, and primary contact person ID.

Parse | Enable SQLCMD | To Notebook

```
1  USE WideWorldImporters
2  SELECT PhoneNumber, PostalCityID,PrimaryContactPersonID
3  FROM Sales.Customers
4
```

Results | Messages

	PhoneNumber	PostalCityID	DeliveryCityID	PrimaryContactPersonID
1	(308) 555-0100	19586	19586	1001
2	(406) 555-0100	33475	33475	1003
3	(480) 555-0100	26483	26483	1005
4	(316) 555-0100	21692	21692	1007
5	(212) 555-0100	12748	12748	1009
6	(701) 555-0100	17054	17054	1011
7	(423) 555-0100	12152	12152	1013
8	(303) 555-0100	3673	3673	1015
9	(201) 555-0100	23805	23805	1017
10	(701) 555-0100	37403	37403	1019
11	(215) 555-0100	8987	8987	1021
12	(218) 555-0100	3081	3081	1023
13	(217) 555-0100	32887	32887	1025
14	(240) 555-0100	19908	19908	1027
15	(210) 555-0100	2111	2111	1029
16	(314) 555-0100	7409	7409	1031

- Retrieve and organize purchase order details from the Purchasing.PurchaseOrders table in the WideWorldImporters database. This query extracts the ContactPersonID, DeliveryMethodID, and ExpectedDeliveryDate, providing a structured view of pending deliveries.

Parse | Enable SQLCMD | To Notebook

```
1  USE WideWorldImporters
2  SELECT ContactPersonID, DeliveryMethodID, ExpectedDeliveryDate
3  FROM Purchasing.PurchaseOrders
4  ORDER BY ExpectedDeliveryDate;
5
```

Results | Messages

	ContactPersonID	DeliveryMethodID	ExpectedDeliveryDate
1	2	9	2013-01-15
2	2	7	2013-01-15
3	2	10	2013-01-15
4	2	2	2013-01-15
5	2	8	2013-01-15
6	2	7	2013-01-15
7	2	7	2013-01-18
8	2	10	2013-01-18
9	2	2	2013-01-18
10	2	8	2013-01-18
11	2	7	2013-01-22
12	2	10	2013-01-22
13	2	2	2013-01-22
14	2	8	2013-01-22
15	2	7	2013-01-22
16	2	7	2013-01-23

Ln 4, Col 32 Spaces: 4 UTF-8 CRLF 2,074 rows MSSQL 00:00:00 localhost,13001 : WideWorldImporters (61)

- Retrieve and analyze purchase order details from the Purchasing.PurchaseOrders table in the WideWorldImporters database. This query extracts the ContactPersonID, DeliveryMethodID, and ExpectedDeliveryDate, sorting the results in descending order of ExpectedDeliveryDate.

Parse | Enable SQLCMD | To Notebook

```
1  USE WideWorldImporters
2  SELECT ContactPersonID, DeliveryMethodID, ExpectedDeliveryDate
3  FROM Purchasing.PurchaseOrders
4  ORDER BY ExpectedDeliveryDate DESC;
5
```

Results | Messages

	ContactPersonID	DeliveryMethodID	ExpectedDeliveryDate
1	2	7	2016-06-20
2	2	2	2016-06-20
3	2	7	2016-06-19
4	2	2	2016-06-19
5	2	7	2016-06-16
6	2	2	2016-06-16
7	2	7	2016-06-15
8	2	2	2016-06-15
9	2	7	2016-06-14
10	2	2	2016-06-14
11	2	7	2016-06-13
12	2	2	2016-06-13
13	2	7	2016-06-12
14	2	2	2016-06-12
15	2	7	2016-06-09
16	2	2	2016-06-09

Ln 4, Col 37 Spaces: 4 UTF-8 CRLF 2,074 rows MSSQL 00:00:00 localhost,13001 : WideWorldImporters (61)

- I’m proposing an analysis of customer transactions from the Sales.CustomerTransactions table in the WideWorldImporters database. Specifically, we are interested in retrieving the CustomerTransactionID, CustomerID, and TransactionAmount fields for transactions where the TransactionAmount exceeds 13,590.

```sql
USE WideWorldImporters
SELECT CustomerTransactionID, CustomerID, TransactionAmount
FROM Sales.CustomerTransactions
WHERE TransactionAmount > 13590;
```

Results | Messages

	CustomerTransactionID	CustomerID	TransactionAmount
1	286	401	13786.20
2	2110	1	14278.29
3	2210	401	13956.40
4	2236	401	17470.80
5	2492	1	19654.65
6	3137	1	14959.66
7	3220	991	17746.80
8	3546	916	23256.45
9	3675	1	18075.70
10	3799	401	19458.00
11	3959	926	18712.80
12	4650	1	14181.80
13	5024	1	16962.50
14	5306	939	16894.65
15	5311	886	17576.60
16	6097	945	20992.10

Ln 4, Col 33 Spaces: 4 UTF-8 CRLF 1,177 rows MSSQL 00:00:00 localhost,13001 : WideWorldImporters (61)

- Retrieve specific order details from the Sales.Orders table in the WideWorldImporters database for CustomerID 77 on May 25, 2015.

```sql
USE WideWorldImporters
SELECT CustomerID, OrderDate
FROM Sales.Orders
WHERE CustomerID = 77 AND OrderDate= '2015-05-25';

```

Results Messages

	CustomerID	OrderDate
1	77	2015-05-25

- The following SQL query retrieves country-specific details from the Application. Countries table within the WideWorldImporters database. The query specifically filters countries located in the continent of Asia and the Western Asia subregion.

Run | Cancel | Disconnect | Change | Database: WideWorldImporters | Estimated Plan

Parse | Enable SQLCMD | To Notebook

```
1   USE WideWorldImporters
2   SELECT CountryID, CountryName,Continent, Subregion
3   FROM  Application.Countries
4   WHERE Continent= 'asia'AND Subregion= 'Western asia';
```

Results | Messages

	CountryID	CountryName	Continent	Subregion
1	12	Armenia	Asia	Western Asia
2	17	Azerbaijan	Asia	Western Asia
3	19	Bahrain	Asia	Western Asia
4	56	Cyprus	Asia	Western Asia
5	80	Georgia	Asia	Western Asia
6	102	Iraq	Asia	Western Asia
7	105	Israel	Asia	Western Asia
8	110	Jordan	Asia	Western Asia
9	116	Kuwait	Asia	Western Asia
1…	120	Lebanon	Asia	Western Asia
1…	162	Oman	Asia	Western Asia
1…	175	Qatar	Asia	Western Asia
1…	186	Saudi Arabia	Asia	Western Asia
1…	212	Syria	Asia	Western Asia
1…	222	Turkey	Asia	Western Asia

- The following SQL query retrieves stock item details from the Warehouse.StockItems table within the WideWorldImporters database. The query specifically filters items based on ColorID = 3 and orders the results in descending order by StockItemID.

Parse | Enable SQLCMD | To Notebook

```
1   USE WideWorldImporters
2   SELECT StockItemID, StockItemName, ColorID
3   FROM Warehouse.StockItems
4   WHERE ColorID = 3
5   ORDER BY StockItemID DESC;
```

Results | Messages

	StockItemID	StockItemName	ColorID
1	174	Bubblewrap dispenser (Black) 1.5m	3
2	133	Furry gorilla with big eyes slippers (Black) XL	3
3	132	Furry gorilla with big eyes slippers (Black) L	3
4	131	Furry gorilla with big eyes slippers (Black) M	3
5	130	Furry gorilla with big eyes slippers (Black) S	3
6	106	Alien officer hoodie (Black) 5XL	3
7	105	Alien officer hoodie (Black) 4XL	3
8	104	Alien officer hoodie (Black) 3XL	3
9	103	Alien officer hoodie (Black) XXL	3
1…	102	Alien officer hoodie (Black) XL	3
1…	101	"The Gu" red shirt XML tag t-shirt (Black) 7XL	3
1…	100	"The Gu" red shirt XML tag t-shirt (Black) 6XL	3
1…	99	"The Gu" red shirt XML tag t-shirt (Black) 5XL	3
1…	98	"The Gu" red shirt XML tag t-shirt (Black) 4XL	3
1…	97	"The Gu" red shirt XML tag t-shirt (Black) 3XL	3
1…	96	"The Gu" red shirt XML tag t-shirt (Black) XXL	3

- The following SQL query retrieves personal details from the Application. People table within the WideWorldImporters database. The query specifically filters records where the FullName is 'Leena De'.

Run | Cancel | Disconnect | Change | Database: WideWorldImporters | Estimated Plan | Ena

Parse | Enable SQLCMD | To Notebook

```
1    USE WideWorldImporters
2    SELECT PersonID, FullName,PhoneNumber, IsPermittedToLogon
3    FROM  Application.People
4    WHERE FullName ='Leena De';
```

Results | Messages

	PersonID	FullName	PhoneNumber	IsPermittedToLogon
1	2066	Leena De	(405) 555-0100	0

- The following SQL query retrieves details about a specific payment method from the Application.PaymentMethods table within the WideWorldImporters database. The query filters records where the PaymentMethodName is 'Cash'.

Parse Enable SQLCMD To Notebook

```
1   USE WideWorldImporters
2   SELECT PaymentMethodName, ValidFrom ,ValidTo, LastEditedBy
3   FROM  Application.PaymentMethods
4   WHERE PaymentMethodName = 'cash';
5
```

Results Messages

	PaymentMethodName	ValidFrom	ValidTo	LastEditedBy
1	Cash	2013-01-01 00:00:00.0000000	9999-12-31 23:59:59.9999999	1

NACE

During this SQL proposition query assignment, I applied a combination of technical expertise and professional skills (NACE) to effectively craft, analyze, and justify each query. Here's how each skill played a role in the process:

1. **Equity & Inclusion:** This skill is crucial in database management, where diverse datasets reflect real-world demographics and business operations.
2. **Career & Self-Development:** Writing these SQL queries reinforced my ability to set professional goals and refine my technical expertise.
3. **Critical Thinking:** SQL queries demand logical structuring and problem-solving abilities. I applied critical thinking to construct efficient queries, ensuring proper filtering, ordering, and aggregation to extract meaningful insights from the database.
4. **Teamwork:** Even in individual SQL assignments, teamwork skills are valuable when collaborating on database management or reporting tasks
5. **Professionalism**: We maintained accountability by following best practices, meeting deadlines, and ensuring accuracy in our work.

THANK YOU