AACS3013 Database Development and Applications

Assignment Submission Form

Programme	DFT
Tutorial Group	DFT1S1G5
Lecturer / Tutor	MR. WANG SHYUN CHAU
Submission Date	8-SEPTEMBER-2024

No.	Student Name	Student ID	Signature	Contribution (%)	Marks
1.	CHONG ZHI YI	24WMD03561	Zy		
2.	LAI JIA TONG	24WMD03620	Jt		
3.	LOW KAI QIN	24WMD03783	kq		
4.	NG JIA HAO	24WMD03590	ng		
5.	PECK JIA CHENG	24WMD03630	JC		
				100	

Feedback / Comments:

AACS3013 Database Development and Applications

DFT

2024

DFT1S1G5

Plagiarism Statement

Read, complete, and sign this statement to be submitted with the written report.

We confirm that the submitted work is all our own work and is in our own words.

	Name (Block Capitals)	Registration No.	Signature
1.	**CHONG ZHI YI**	24WMD03561	....Zy........
2.	**LAI JIA TONG**	24WMD03620	....Jt........
3.	**LOW KAI QIN**	24WMD03783	....kq........
4.	**NG JIA HAO**	24WMD03590	....ng........
5.	**PECK JIA CHENG**	24WMD03630	....JC........

Tutorial Group : **DFT1S1G5**

Date : **8-SEPTEMBER-2024**

Assignment Assessment Rubrics

Programme: DFT **Tutorial Group : DFT1S1G5**

Criteria	**Marks**	**Excellent**	**Good**	**Average**	**Poor**	**Score**
GROUP						
(1) Organisation background	10	Very high level organisational analysis and research, covering all relevant details. Very clear and excellent description of problem statement and solution. 9 – 10	High level organisational analysis and research, covering mostly relevant details. Good and clear description of problem statement and solution. 6 – 8	Some organisational analysis and research, barely covering relevant details. Acceptable and clear description of problem statement and solution. 3 – 5	Lack of organisational analysis and research, covering details mostly not relevant. Insufficient and unclear description of problem statement and solution. 0 – 2	
(2) Business rules	10	Excellent business rules. 9 – 10	Good business rules. 6 – 8	Acceptable business rules. 3 – 5	Poor or incorrect business rules. 0 – 2	
(3) Database Design (ERD)	10	Excellent database design. 9 – 10	Good database design. 6 – 8	Acceptable database design. 3 – 5	Poor database design. 0 – 2	
(4) Create database tables in Oracle	20	Excellent design of Oracle database with all requirements met. 17 – 20	Good design of Oracle database with major requirements met. 11 – 16	Acceptable design of Oracle database with some requirements met. 5 – 10	Poor design of Oracle database and does not meet most of the requirements shown. 0 – 4	
(5) Create records	20	Excellent and sufficient records resemble real world data. 17 – 20	Good and sufficient records resemble real world data. 11 – 16	Acceptable and some adequate records resemble real world data. 5 – 10	Poor and unrealistic data. 0 – 4	
(7) Assignment report	10	Excellent writing style. Messages and answers are clear and effective. Full compliance of format. 9 – 10	Good writing style. Messages and answers sometimes lack clarity. Close compliance of format. 6 – 8	Average writing style. Messages and answers are poorly conveyed. Minimum compliance of format. 3 – 5	Writing style is poor. Answers are poorly conveyed. Non-compliance of format. 0 – 2	

Total	80					
INDIVIDUAL						
(6) Create query	10	Excellent and correct design of query statement based on the purpose/ scenario with excellent formatting. 9 – 10	Good and correct design of query statement based on the purpose/ scenario with appropriate formatting. 6 – 8	Average and correct design query statement based on the purpose/ scenario with appropriate formatting 3 – 5	Poor and incorrect design query statement with poor formatting. 0 – 2	
(8) Presentation	10	Excellent communication skills. Messages and answers are clear and effective. 9 – 10	Good communication skills. Messages and answers sometimes lack clarity. 6 – 8	Average communication skill. Messages and answers are poorly conveyed. 3 – 5	Use of language is ineffective; communication is generally poor. Answers are poorly conveyed. 0 – 2	
Total	20					

Name	**Query (10%)**	**Presentation (10%)**	**Marks (Task 6 & 8)**	**Remark (if any)**
1. CHONG ZHI YI				
2. LAI JIA TONG				
3. LOW KAI QIN				
4. NG JIA HAO				
5. PECK JIA CHENG				

TABLE OF CONTENT

REPORT CONTENT

Task 1: Organization Background

Founded in 1993, NVIDIA has significantly influenced technology with its introduction of the Graphics Processing Unit (GPU) in 1999, which revolutionized PC gaming and laid the groundwork for modern artificial intelligence (AI). Over the years, NVIDIA has expanded beyond GPUs into Central Processing Units (CPUs), Data Processing Units (DPUs), and AI software, establishing itself as a leader in full-stack computing. The company's GPUs are vital for high-performance gaming and complex visual tasks, while its AI technologies drive advancements in machine learning and deep learning. NVIDIA's impact extends across various industries: it powers autonomous vehicle data centers, supports over 25,000 AI-driven companies, engages 150,000 users with its Omniverse platform for digital twins, serves 200 million gamers with GeForce GPUs, aids 400,000 developers in healthcare with the MONAI framework, and supports 1 million developers in robotics with the Jetson platform.

Despite these advancements, NVIDIA is facing significant challenges in managing its sales and customer data due to the use of fragmented systems and manual processes. These issues include data redundancy, where duplicate and inconsistent records across different systems create inefficiencies in tracking sales and managing customer interactions. The dispersed nature of the data also complicates the generation of accurate reports and the performance of insightful analyses, impacting decision-making and overall operational performance.

Additionally, the current data management approach lacks centralized security measures, leaving sensitive information vulnerable to unauthorized access and potential breaches. As NVIDIA grows, the existing systems struggle to handle increasing data volumes, leading to scalability issues and a higher risk of operational errors.

To address these challenges, NVIDIA should implement a centralized database system. This solution will consolidate all sales, customer, and inventory data into a unified platform, effectively eliminating data redundancy and ensuring consistency across records. A centralized database will streamline data management processes, making it easier to generate comprehensive reports and conduct detailed analyses. It will also enhance data security by incorporating advanced measures such as encryption and access controls, protecting sensitive information from unauthorized access and breaches. Furthermore, the system will support scalability, efficiently handling growing data volumes and reducing the risk of operational issues as NVIDIA continues to expand.

In summary, adopting a centralized database will resolve current issues related to data redundancy, management inefficiencies, security vulnerabilities, and scalability challenges, thereby improving overall operational efficiency and supporting NVIDIA’s continued growth and success.

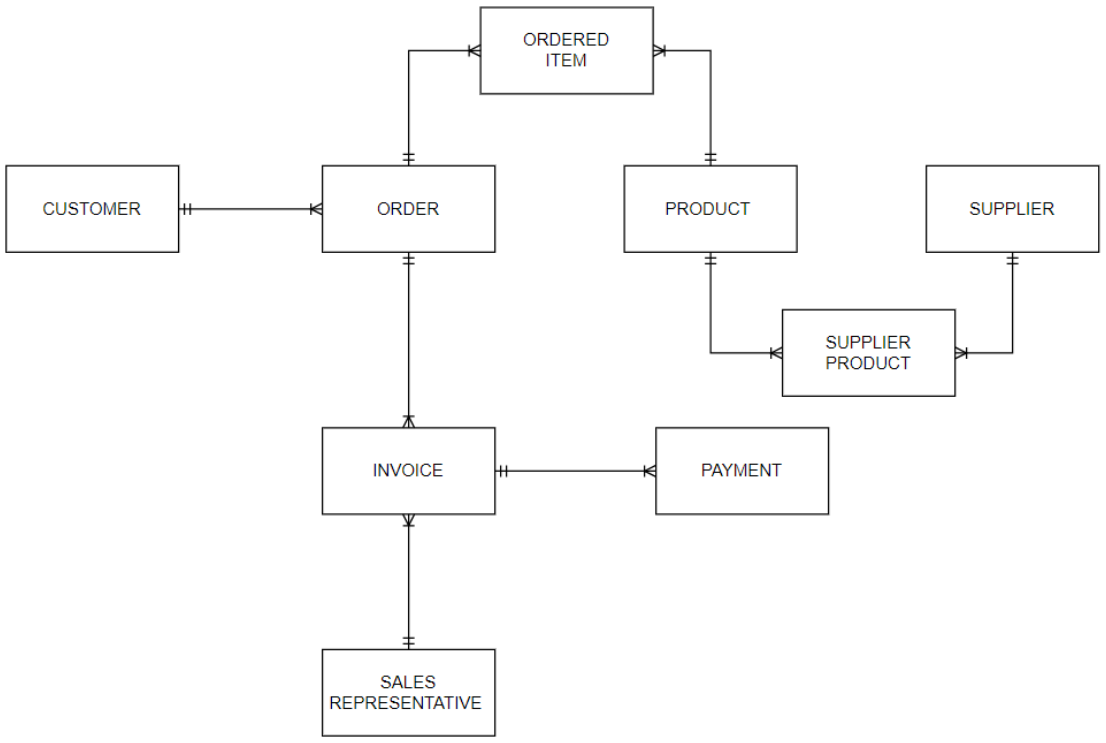
Task 2: Develop Business Rules

Business Rule

1. Each **Customer** can place one or many **Order**. Each **Order** is associated with only one **Customer**.
2. Each **Order** can include one or many **Product**. Each **Product** can be included in one or many **Order**.
3. Each **Product** may be supplied by one or many **Supplier**. Each **Supplier** can provide one or many **Product**.
4. Each **Order** can be associated with one or many **Invoice**. Each **Invoice** is linked to only one **Order**.
5. Each **Invoice** is assigned to one **Sales Representative**. Each **Sales Representative** can manage many **Invoice**.
6. Each **Invoice** can generate one or many **Payment**. Each **Payment** is linked to only one **Invoice**.

Task 3: Develop Business Rules

3.1 Entity-Relationship Diagram



3.2 Database Design Language

CUSTOMER (<u>customerID</u> ,name,address,phoneNo,email)

ORDERS (<u>orderID,</u>customerID*,orderDate,orderStatus,totalAmount,SST)

ORDERED_ITEM
(<u>orderID</u>*,<u>productID</u>*,unitPrice,orderQuantity,lineTotalBeforeDisc,Discount,totalAfterDisc)

PRODUCT (<u>productID</u>,productName,description,brand,category,quantity,location,cost,sellingPrice)

SUPPLIER_PRODUCT
(<u>productID*</u>,<u>supplierNo*</u>,fulfillmentDate,fulfillmentTerms,supplyPrice,minOrderQty,defaultSupplier)

SUPPLIER (<u>supplierNo</u>,PIC,companyName,contactNo,address,country)

INVOICE (<u>InvoiceNo</u>,agentID*,orderID*,date,currency,amount,terms)

PAYMENT (<u>paymentID</u>,InvoiceNo*,date,amount,paymentMethod,referenceNo)

SALES_REPRESENTATIVE (<u>agentID</u>,name,salary,salesTarget,commissionPercentage)

Task 4: Create Database Table in Oracle

4.1 CUSTOMER

```
CREATE TABLE CUSTOMER (
        CUSTOMERIDVARCHAR (8) NOT NULL,
        NAME          VARCHAR (25) NOT NULL,
        ADDRESS       VARCHAR (50),
        PHONENO       VARCHAR (14),
        EMAIL         VARCHAR (35),
        PRIMARY KEY(CUSTOMERID),
        CONSTRAINT CHK_EMAIL CHECK (REGEXP_LIKE(email, '^[a-zA-Z]\w+@\S+\.\S+$'))
);
```

4.2 ORDERS

```
CREATE TABLE ORDERS (
        ORDERID             VARCHAR (5) NOT NULL,
        ORDERDATE           DATE NOT NULL,
        ORDERSTATUS         VARCHAR (9),
        TOTALAMOUNT         NUMBER (10, 2),
        SST                 NUMBER (9, 2),
        CUSTOMERID          VARCHAR (8) NOT NULL,
        PRIMARY KEY (ORDERID),
        FOREIGN KEY (CUSTOMERID) REFERENCES CUSTOMER (CUSTOMERID),
        CONSTRAINT CHECK_ORDERSTATUS CHECK (ORDERSTATUS IN ('PENDING',
        'SHIPPED', 'DELIVERED')),
        CONSTRAINT CHECK_TOTALAMOUNT CHECK (TOTALAMOUNT > 0)
);
```

4.3 ORDERED_ITEM

```
CREATE TABLE ORDERED_ITEM (
	ORDERID			VARCHAR (5) NOT NULL,
	PRODUCTID		VARCHAR (5) NOT NULL,
	ORDERQUANTITY		NUMBER (7) NOT NULL,
	UNITPRICE		NUMBER (10,2) NOT NULL,
	LINETOTALBEFOREDISC	NUMBER (12,2),
	DISCOUNT		NUMBER (3),
	TOTALAFTERDISC		NUMBER (12,2),
	PRIMARY KEY (ORDERID, PRODUCTID),
	FOREIGN KEY (ORDERID) REFERENCES ORDERS (ORDERID),
	FOREIGN KEY (PRODUCTID) REFERENCES PRODUCT (PRODUCTID),
	CONSTRAINT CHECK_UNITPRICE CHECK (UNITPRICE > 0),
	CONSTRAINT CHECK_ORDERQUANTITY CHECK (ORDERQUANTITY > 0),
	CONSTRAINT CHECK_DISCOUNT CHECK (DISCOUNT >= 0 AND DISCOUNT <= 100)
);
```

4.4 PRODUCT

```
CREATE TABLE PRODUCT (
	PRODUCTID		VARCHAR (5) NOT NULL,
	PRODUCTNAME		VARCHAR (40) NOT NULL,
	DESCRIPTION		VARCHAR (60),
	BRAND			VARCHAR (20),
	CATEGORY		VARCHAR (26),
	QUANTITY		NUMBER (5),
	LOCATION		VARCHAR (50),
	COST			NUMBER (7,2) NOT NULL,
	SELLINGPRICE		NUMBER (8,2),
	PRIMARY KEY (PRODUCTID),
	CONSTRAINT CHECK_CATEGORY CHECK (CATEGORY IN ('H-LAPTOPS AND
	WORKSTATIONS', 'H-NETWORKING', 'H-GPUS', 'S-APPS AND TOOLS', 'S-
	INFRASTRUCTURE', 'S-CLOUD SERVICES')),
	CONSTRAINT CHECK_QUANTITYPRODUCT CHECK (QUANTITY > 0),
	CONSTRAINT CHECK_COST CHECK (COST > 0),
	CONSTRAINT CHECK_SELLINGPRICE CHECK (SELLINGPRICE > 0 AND
	SELLINGPRICE >= COST)
);
```

4.5 SUPPLIER_PRODUCT

```
CREATE TABLE SUPPLIER_PRODUCT (
        PRODUCTID               VARCHAR (5) NOT NULL,
        SUPPLIERNO              NUMBER (4) NOT NULL,
        FULFILLMENTDATE         DATE,
        FULFILLMENTTERMS        VARCHAR (7),
        SUPPLYPRICE             NUMBER (10,2) NOT NULL,
        MOQ                     NUMBER (5) NOT NULL,
        DEFAULTSUPPLIER         VARCHAR (3),
        PRIMARY KEY (PRODUCTID, SUPPLIERNO),
        FOREIGN KEY (PRODUCTID) REFERENCES PRODUCT (PRODUCTID),
        FOREIGN KEY (SUPPLIERNO) REFERENCES SUPPLIER (SUPPLIERNO),
        CONSTRAINT CHECK_PRICE CHECK (SUPPLYPRICE >0),
        CONSTRAINT CHECK_MOQ CHECK (MOQ >= 0),
        CONSTRAINT CHECK_FULFILLMENTTERMS CHECK (FULFILLMENTTERMS IN
        ('NET30', 'NET60', 'NET90', 'COD', 'PREPAID')),
        CONSTRAINT CHECK_DEFAULTSUPPLIER CHECK (DEFAULTSUPPLIER IN ('YES',
        'NO'))
);
```

4.6 SUPPLIER

```
CREATE TABLE SUPPLIER (
        SUPPLIERNO  NUMBER (4) NOT NULL,
        PIC         VARCHAR (25),
        CONTACTNO NUMBER (14)NOT NULL,
        EMAIL       VARCHAR (35),
        ADDRESS     VARCHAR (50),
        COUNTRY     VARCHAR (10),
        PRIMARY KEY(SUPPLIERNO),
        CONSTRAINT CHK_EMAILSUPPLIER CHECK (REGEXP_LIKE(email,'^[a-zA-
        Z]\w+@(\S+)$'))
);
```

4.7 INVOICE

```
CREATE TABLE INVOICE (
        INVOICENO           VARCHAR (13) NOT NULL,
        DATE_INVOICE        DATE,
        CURRENCY            VARCHAR (3),
        QUANTITY            NUMBER (4),
        AMOUNT              NUMBER (10, 2),
        TERMS               VARCHAR (7),
        ORDERID             VARCHAR (5) NOT NULL,
        AGENTID             VARCHAR (8) NOT NULL,
        PRIMARY KEY (INVOICENO),
        FOREIGN KEY (ORDERID) REFERENCES ORDERS(ORDERID),
        FOREIGN KEY (AGENTID) REFERENCES SALES_REPRESENTATIVE(AGENTID),
        CONSTRAINT CHECK_TERMS CHECK (TERMS IN ('NET30', 'NET60', 'NET90', 'COD',
        'PREPAID')),
        CONSTRAINT CHECK_QUANTITY CHECK (QUANTITY > 0),
        CONSTRAINT CHECK_AMOUNT CHECK (AMOUNT > 0)
);
```

4.8 PAYMENT

```
CREATE TABLE PAYMENT (
        PAYMENTID           VARCHAR (7) NOT NULL,
        PAYMENTDATE         DATE NOT NULL,
        AMOUNT              NUMBER (12,2) NOT NULL,
        PAYMENTMETHOD       VARCHAR (5) NOT NULL,
        REFERENCENO         NUMBER (10) NOT NULL,
        INVOICENO           VARCHAR (13) NOT NULL,
        PRIMARY KEY (PAYMENTID),
        FOREIGN KEY (INVOICENO) REFERENCES INVOICE (INVOICENO),
        CONSTRAINTS CHK_AMOUNT CHECK (AMOUNT > 0),
        CONSTRAINT CHK_PAYMENTMETHOD CHECK (PAYMENTMETHOD IN ('CASH',
        'TNG', 'PPB', 'MBB', 'CIMB'))
);
```

4.9 SALES_REPRESENTATIVE

```
CREATE TABLE SALES_REPRESENTATIVE (
        AGENTID                     VARCHAR (8) NOT NULL,
        NAME                        VARCHAR (20) NOT NULL,
        SALARY                      NUMBER (10,2),
        SALESTARGET                 NUMBER (10,2),
        COMMISSIONPERCENTAGE NUMBER (3,1),
        PRIMARY KEY (AGENTID),
        CONSTRAINT CHECK_COMMISSIONPERCENTAGE CHECK
        (COMMISSIONPERCENTAGE >= 0 AND COMMISSIONPERCENTAGE <= 10)
);
```

Task 5: Create Records

5.1 CUSTOMER

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0001', 'Sheelah Oldall', '8 Dayton Road', '683 991 3860', 'soldall0@nyu.edu');

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0002', 'Loy Heaviside', '0323 Village Green Street', '310 960 5823', 'lheaviside1@phpbb.com');

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0003', 'Walt Greggs', '70698 Graedel Terrace', '635 289 1088', 'wgreggs2@china.com.cn');

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0004', 'Hillel Chaize', '68650 Sunfield Pass', '397 467 7004', 'hchaize3@java.com');

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0005', 'Marjy Francie', '2 Onsgard Alley', '748 719 0339', 'mfrancie4@wired.com');

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0006', 'Trudi Ellwood', '7596 Sutteridge Trail', '171 796 8244', 'tellwood5@symantec.com');

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0007', 'Della Dobey', '55 Browning Alley', '530 883 1231', 'ddobey6@chicagotribune.com');

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0008', 'Stefano Aloshkin', '71613 Bartelt Park', '519 627 7182', 'saloshkin7@google.co.uk');

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0009', 'Si Darton', '1827 Atwood Park', '231 329 1517', 'sdarton8@netlog.com');

INSERT INTO CUSTOMER (customerID, name, address, phoneNo, email) VALUES ('CS0010', 'Terrye Riguard', '1607 International Way', '759 868 3652', 'triguard9@hatena.ne.jp');

5.2 ORDERS

```
INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR001', '30-Jun-24', 'Shipped', '4027919.25 ', '241675.16 ', 'CS0001');

INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR002', '24-Jul-22', 'Delivered', '1093353.83 ', '65601.23 ', 'CS0002');

INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR003', '8-Jul-24', 'Delivered', '9713531.74 ', '582811.90 ', 'CS0003');

INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR004', '22-Mar-23', 'Pending', '8895299.07 ', '533717.94 ', 'CS0004');

INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR005', '13-Sep-22', 'Delivered', '5548883.66 ', '332933.02 ', 'CS0005');

INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR006', '25-May-23', 'Pending', '9593044.56 ', '575582.67 ', 'CS0006');

INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR007', '22-Feb-23', 'Shipped', '3846688.84 ', '230801.33 ', 'CS0007');

INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR008', '14-Dec-22', 'Pending', '3560993.30 ', '213659.60 ', 'CS0008');

INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR009', '6-May-24', 'Shipped', '8501908.95 ', '510114.54 ', 'CS0009');

INSERT INTO ORDERS (orderID, orderDate, orderStatus, totalAmount, SST, customerID) VALUES
('OR010', '20-Feb-24', 'Pending', '9346121.90 ', '560767.31 ', 'CS0010');
```

5.3 ORDERED_ITEM

```sql
INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR001', 'PD001', '79', '4412676.27', '348601425.3', '41',
'205674840.9');

INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR001', 'PD002', '9', '8655683.84', '77901154.56', '68',
'24928369.46');

INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR001', 'PD003', '75', '2353166.49', '176487486.8', '38',
'109422241.8');

INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR002', 'PD004', '8', '282525.05', '2260200.4', '79', '474642.08');

INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR002', 'PD005', '48', '8403990.42', '403391540.2', '79',
'84712223.44');

INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR002', 'PD006', '80', '5883218.63', '470657490.4', '24',
'357699692.7');

INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR002', 'PD007', '92', '4424581.79', '407061524.7', '65',
'142471533.7');

INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR003', 'PD008', '73', '9053917.67', '660935989.9', '64',
'237936956.4');

INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR003', 'PD009', '68', '2098260.22', '142681695', '82', '25682705.1');

INSERT INTO ORDERED_ITEM (orderID, productID, orderQuantity, unitPrice, lineTotalBeforeDisc,
discount, totalAfterDisc) VALUES ('OR004', 'PD010', '62', '7701711.52', '477506114.2', '26',
'353354524.5');
```

5.4 PRODUCT

```sql
INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD001', 'NVIDIA A10 SXM4', 'High-performance GPU with 24GB
VRAM for gaming', 'Boxx Technologies', 'H-Laptops and Workstations', 15, 'Warehouse A', 6209.65,
420218.33);

INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD002', 'NVIDIA Quadro FX 5800 8GB', 'Versatile GPU with 8GB
VRAM for gaming and creation', 'Alienware (Dell)', 'H-GPUs', 32, 'Store 1', 93476.47, 566919.82);

INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD003', 'NVIDIA Quadro P6000 Max-Q 24GB', 'Mid-range GPU with
24GB VRAM for intensive tasks', 'Alienware (Dell)', 'H-Laptops and Workstations', 34, 'Warehouse B',
69066.29, 533038.82);

INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD004', 'NVIDIA GeForce GTX 750 Ti 2GB', 'Efficient GPU with 8GB
VRAM for basic gaming', 'Acer', 'S-Apps and Tools', 28, 'Store 2', 39511.8, 136632.57);

INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD005', 'NVIDIA Quadro RTX 6000 Max-Q', 'Efficient GPU with 8GB
VRAM for intensive tasks', 'Acer', 'S-Apps and Tools', 69, 'Store 1', 72076.65, 542339.96);

INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD006', 'NVIDIA Quadro RTX 8000 SLI', 'Advanced GPU with 10GB
VRAM for various tasks', 'Newegg', 'H-Networking', 106, 'Store 1', 7823.61, 78201.44);

INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD007', 'NVIDIA Tesla K80 SXM4', 'Mid-range GPU with 8GB VRAM
for light use', 'ASUS', 'H-GPUs', 43, 'Store 2', 85562.23, 519164.4);

INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD008', 'NVIDIA GeForce GTX 780 3GB', 'High-performance GPU with
24GB VRAM for gaming', 'Acer', 'S-Cloud Services', 61, 'Warehouse B', 18283.27, 963266.77);

INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD009', 'NVIDIA Quadro P4000 SLI', 'High-end GPU with 8GB VRAM
for intensive gaming', 'Acer', 'S-Apps and Tools', 90, 'Warehouse A', 82808.91, 237493.36);

INSERT INTO PRODUCT (productId, productName, description, brand, category, quantity, location,
cost, sellingPrice) VALUES ('PD010', 'NVIDIA Tesla K20 PCIe', 'High-end GPU with 24GB VRAM for
high-end tasks', 'Amazon', 'S-Infrastructure', 19, 'Warehouse A', 55197.34, 633882.53);
```

5.5 SUPPLIER_PRODUCT

```
INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD001', '1001', '13-Jun-21 ', 'COD', '660636.98', '37',
'NO');

INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD002', '1002', '23-Jun-21 ', 'NET60', '943297.98', '63',
'NO');

INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD003', '1003', '4-Jan-24 ', 'PREPAID', '532588.65', '3',
'NO');

INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD004', '1004', '13-Mar-21 ', 'NET30', '609542.69', '78',
'YES');

INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD005', '1005', '18-Sep-23 ', 'COD', '170088.72', '85',
'YES');

INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD006', '1006', '25-Aug-23 ', 'NET60', '803770.25', '74',
'YES');

INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD007', '1007', '25-Jun-22 ', 'NET60', '834486.25', '88',
'NO');

INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD008', '1008', '28-Aug-23 ', 'NET30', '817855.53', '12',
'NO');

INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD009', '1009', '20-Mar-23 ', 'NET30', '438165.39', '2',
'NO');

INSERT INTO SUPPLIER_PRODUCT (productID, supplierNo, fulfillmentDate , fulfillmentTerms,
supplyPrice, MOQ, defaultSupplier) VALUES ('PD010', '1010', '1-Feb-23 ', 'NET60', '997202.46', '43',
'YES');
```

5.6 SUPPLIER

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1001, 'Sheena Collicott', '2379296585', 'scollicott0@blogtalkradio.com', '3 Victoria Center', 'Malaysia');

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1002, 'Leonelle Ebbs', '6866421734', 'lebbs1@infoseek.co.jp', '3 Vera Drive', 'Malaysia');

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1003, 'Saw Brunger', '6285430219', 'sbrunger2@joomla.org', '6 Mayer Road', 'Malaysia');

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1004, 'Agnesse Leban', '5924486802', 'aleban3@businessinsider.com', '4 Reinke Avenue', 'Malaysia');

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1005, 'Jessamyn Happs', '4346723584', 'jhapps4@illinois.edu', '20712 Esker Plaza', 'Malaysia');

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1006, 'Christa Cridlan', '8295811352', 'ccridlan5@ucoz.com', '228 Tennessee Avenue', 'Malaysia');

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1007, 'Seumas Coraini', '5309044528', 'scoraini6@google.es', '983 Elka Alley', 'Malaysia');

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1008, 'Kevon Quarterman', '4606828254', 'kquarterman7@etsy.com', '87695 Village Green Place', 'Malaysia');

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1009, 'Carlos Wein', '1967426299', 'cwein8@meetup.com', '32097 Bunker Hill Center', 'Malaysia');

INSERT INTO SUPPLIER (supplierNo, PIC, contactNo, email, address, country) VALUES (1010, 'Brocky Kapelhof', '1997968518', 'bkapelhof9@go.com', '06547 Eliot Street', 'Malaysia');

5.7 INVOICE

```
INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0001', '8-Feb-22 ', 'MYR', '661', '37938412.33', 'COD', 'OR217',
'AGT181');

INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0002', '8-Jan-22 ', 'MYR', '63', '53270844.03', 'COD', 'OR196',
'AGT236');

INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0003', '5-Jun-22 ', 'MYR', '420', '51922628.57', 'COD', 'OR006',
'AGT127');

INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0004', '7-Apr-22 ', 'MYR', '349', '61498098.05', 'NET60', 'OR003',
'AGT227');

INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0005', '9-Jul-22 ', 'MYR', '110', '90737242.79', 'PREPAID', 'OR041',
'AGT043');

INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0006', '28-Dec-22 ', 'MYR', '883', '85038478.03', 'PREPAID', 'OR185',
'AGT094');

INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0007', '27-Aug-22 ', 'MYR', '17', '82683535.06', 'PREPAID', 'OR116',
'AGT086');

INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0008', '23-Oct-22 ', 'MYR', '622', '37366226.55', 'NET90', 'OR061',
'AGT156');

INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0009', '16-Apr-22 ', 'MYR', '819', '991960.44', 'NET60', 'OR202',
'AGT049');

INSERT INTO INVOICE (InvoiceNo, date_invoice , currency, quantity, amount, terms, orderID,
agentID) VALUES ('INV-2022/0010', '15-Jan-22 ', 'MYR', '469', '50931812.16', 'PREPAID', 'OR019',
'AGT223');
```

5.8 PAYMENT

```
INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0001', '20-Feb-2022', '7246106028.18 ', '763422390', 'PPB', 'INV-
2022/0001');

INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0002', '07-Nov-2022', '7669509906.83 ', '560543500', 'CASH', 'INV-
2022/0002');

INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0003', '18-Jun-2022', '3131540200.84 ', '735490098', 'TNG', 'INV-
2022/0003');

INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0004', '25-Aug-2022', '5747643712.08 ', '68130039', 'MBB', 'INV-
2022/0004');

INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0005', '15-Oct-2022', '67672627.58 ', '464221923', 'MBB', 'INV-2022/0005');

INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0006', '20-Jun-2022', '3024649434.80 ', '622422915', 'TNG', 'INV-
2022/0006');

INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0007', '30-Mar-2022', '7644210346.69 ', '736916455', 'PPB', 'INV-
2022/0007');

INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0008', '12-Nov-2022', '9551399250.39 ', '448937021', 'CIMB', 'INV-
2022/0008');

INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0009', '11-Feb-2022', '9030803368.69 ', '377460798', 'CIMB', 'INV-
2022/0009');

INSERT INTO PAYMENT (paymentID, paymentDate, amount, referenceNo, paymentMethod,
InvoiceNo) VALUES ('PY-0010', '13-Aug-2022', '2346321200.88 ', '182569049', 'TNG', 'INV-
2022/0010');
```

5.9 SALES_REPRESENTATIVE

```
INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT001', 'Candace Moogan', '1800', '28879 ', '6.55 ');

INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT002', 'Saxon Sousa', '1800', '18688 ', '7.86 ');

INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT003', 'Gris Strowger', '3900', '884853 ', '5.89 ');

INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT004', 'Hannis Lawrance', '6000', '196334 ', '3.1 ');

INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT005', 'Fancie Eastwood', '3500', '450379 ', '3.13 ');

INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT006', 'Sybila Clemas', '6000', '239021 ', '3.23 ');

INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT007', 'Willyt Moniker', '4000', '357349 ', '8.64 ');

INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT008', 'Sharlene Middlemiss', '3400', '837238 ', '1.88 ');

INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT009', 'Lexy Colquit', '5000', '699495 ', '3.3 ');

INSERT INTO SALES_REPRESENTATIVE (agentID, name, salary, salesTarget ,
commissionPercentage ) VALUES ('AGT010', 'Gawen Rainbird', '5000', '850717 ', '8.55 ');
```

Task 6: Create Query

6.1 Query Total Sales by Each Sales Representative (LAI JIA TONG)

1. Purpose:

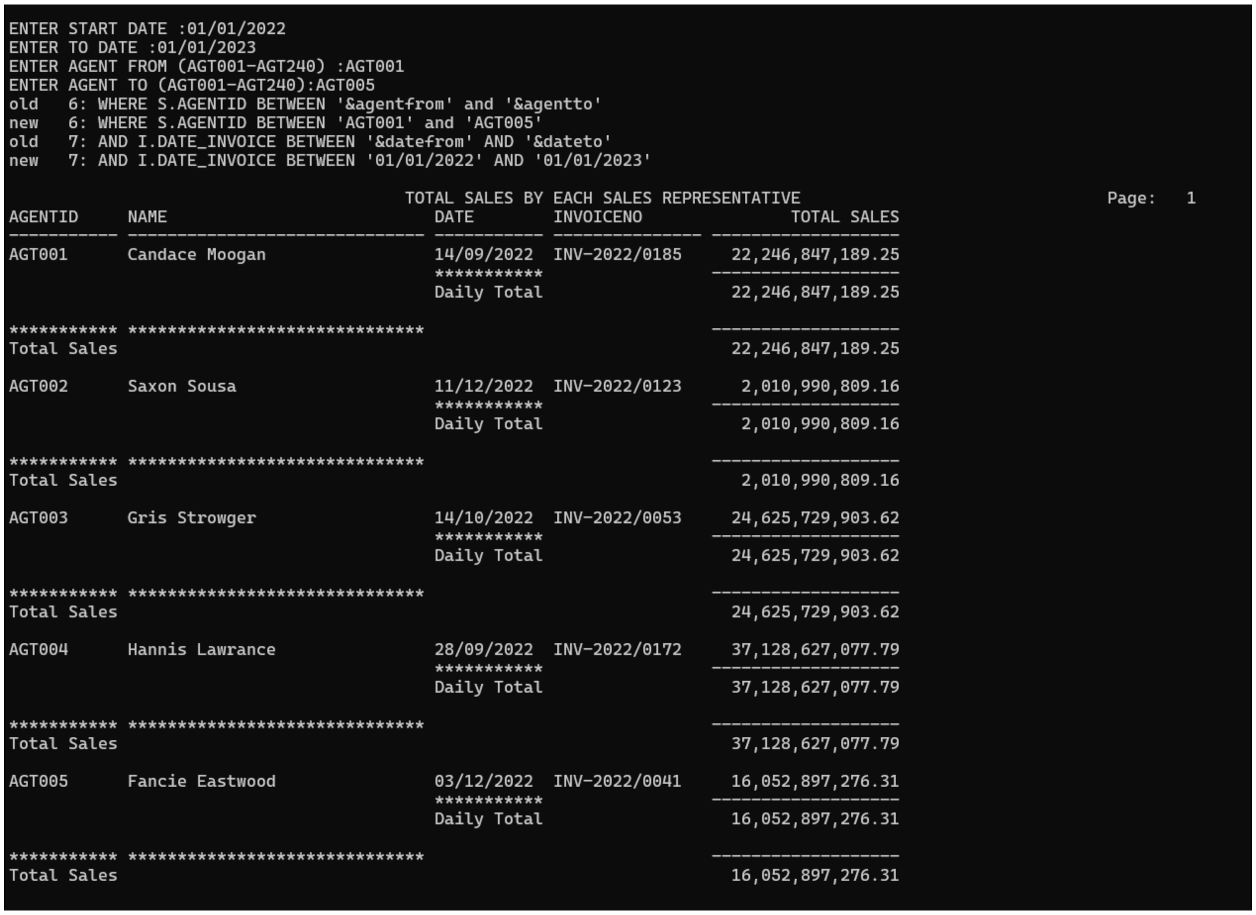
By querying the total sales for each sales representative and specifying the date range, businesses can effectively evaluate performance. This analysis enables companies to identify top performers and address any underperformance. Additionally, examining historical sales data allows businesses to forecast future sales and adjust their strategies accordingly. Sales data also plays a crucial role in determining commissions, bonuses, and other incentives for sales staff, ensuring that compensation aligns with their achievements.

2. Query:

```
ALTER SESSION SET NLS_DATE_FORMAT = 'dd/mm/yyyy';

ACCEPT datefrom DATE FORMAT 'dd/mm/yyyy' DEFAULT '01/01/2022' PROMPT 'ENTER START
DATE :'
ACCEPT dateto DATE FORMAT 'dd/mm/yyyy' DEFAULT '28/08/2024' PROMPT 'ENTER TO
DATE :'

ACCEPT agentfrom CHAR PROMPT 'ENTER AGENT FROM (AGT001-AGT240) :'
ACCEPT agentto CHAR PROMPT 'ENTER AGENT TO (AGT001-AGT240):'

COLUMN AGENTID FORMAT A11
COLUMN NAME FORMAT A30
COLUMN "TOTAL SALES" FORMAT 999,999,999,990.99
COLUMN "DATE" FORMAT A11
COLUMN INVOICENO FORMAT A15

SET LINESIZE 120
SET PAGESIZE 60
TTITLE CENTER 'TOTAL SALES BY EACH SALES REPRESENTATIVE' RIGHT 'Page:' FORMAT
999 SQL.PNO SKIP 1

BREAK ON AGENTID SKIP 1 ON NAME ON "DATE" SKIP 1
COMPUTE SUM LABEL 'Daily Total' of "TOTAL SALES" ON "DATE"
COMPUTE SUM LABEL 'Total Sales' of "TOTAL SALES" ON AGENTID

SELECT
S.AGENTID,S.NAME,I.DATE_INVOICE AS "DATE",I.INVOICENO,
I.QUANTITY*I.AMOUNT AS "TOTAL SALES"
FROM SALES_REPRESENTATIVE S
JOIN INVOICE I ON S.AGENTID = I.AGENTID
WHERE S.AGENTID BETWEEN '&agentfrom' and '&agentto'
AND I.DATE_INVOICE BETWEEN '&datefrom' AND '&dateto'
ORDER BY S.AGENTID ASC;
```

CLEAR COLUMNS
TTITLE OFF

3. Sample Output:



```
ENTER START DATE :01/01/2022
ENTER TO DATE :01/01/2023
ENTER AGENT FROM (AGT001-AGT240) :AGT001
ENTER AGENT TO (AGT001-AGT240):AGT005
old   6: WHERE S.AGENTID BETWEEN '&agentfrom' and '&agentto'
new   6: WHERE S.AGENTID BETWEEN 'AGT001' and 'AGT005'
old   7: AND I.DATE_INVOICE BETWEEN '&datefrom' AND '&dateto'
new   7: AND I.DATE_INVOICE BETWEEN '01/01/2022' AND '01/01/2023'

                                        TOTAL SALES BY EACH SALES REPRESENTATIVE                               Page:   1
AGENTID     NAME                           DATE        INVOICENO               TOTAL SALES
----------- ------------------------------ ----------- --------------- -------------------
AGT001      Candace Moogan                 14/09/2022  INV-2022/0185     22,246,847,189.25
                                           ***********                 -------------------
                                           Daily Total                   22,246,847,189.25

*********** ******************************                             -------------------
Total Sales                                                              22,246,847,189.25

AGT002      Saxon Sousa                    11/12/2022  INV-2022/0123      2,010,990,809.16
                                           ***********                 -------------------
                                           Daily Total                    2,010,990,809.16

*********** ******************************                             -------------------
Total Sales                                                               2,010,990,809.16

AGT003      Gris Strowger                  14/10/2022  INV-2022/0053     24,625,729,903.62
                                           ***********                 -------------------
                                           Daily Total                   24,625,729,903.62

*********** ******************************                             -------------------
Total Sales                                                              24,625,729,903.62

AGT004      Hannis Lawrance                28/09/2022  INV-2022/0172     37,128,627,077.79
                                           ***********                 -------------------
                                           Daily Total                   37,128,627,077.79

*********** ******************************                             -------------------
Total Sales                                                              37,128,627,077.79

AGT005      Fancie Eastwood                03/12/2022  INV-2022/0041     16,052,897,276.31
                                           ***********                 -------------------
                                           Daily Total                   16,052,897,276.31

*********** ******************************                             -------------------
Total Sales                                                              16,052,897,276.31
```

6.2 Query Top 10 Selling Product (CHONG ZHI YI)

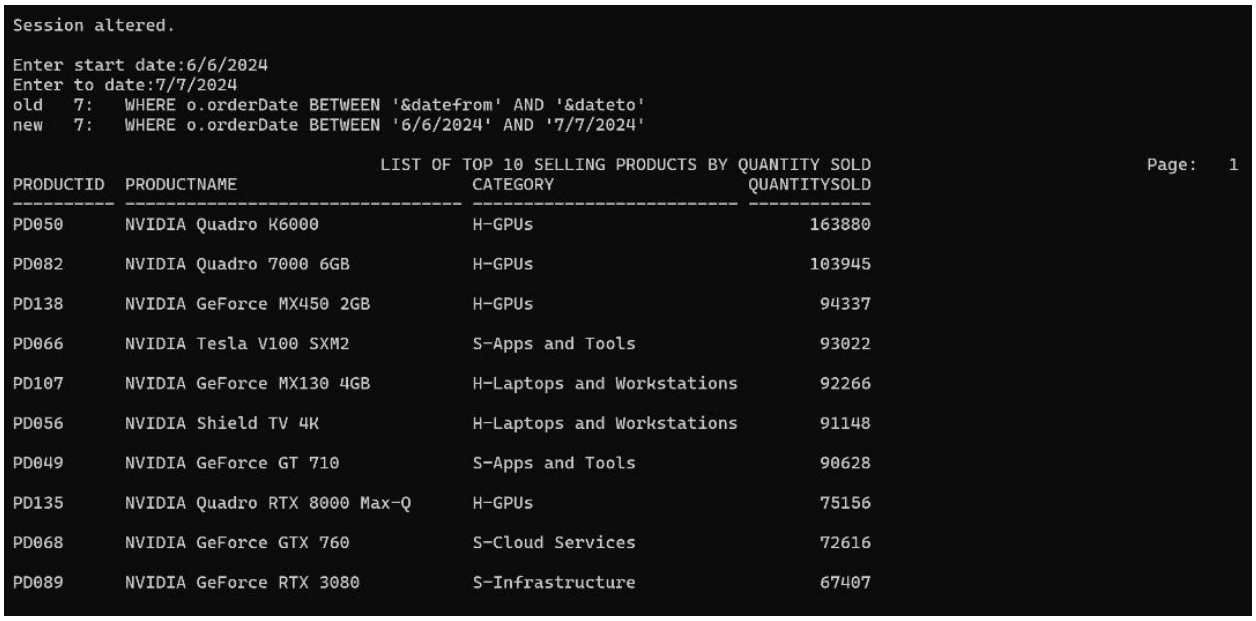
1. Purpose:

The purpose is to identify the top 10 selling products by quantity sold within a specified date range. This query aims to analyze and rank products based on their sales volume to determine which products are the most popular or have the highest sales during the given period. Therefore, this helps in identifying the best-performing products, understanding sales trends, and making informed decisions about inventory, marketing, and product strategies. It involves inventory management, targeting the appropriate market, and focusing on profitable product categories to boost sales and meet customers' needs.

2. Query:

```
Alter session set nls_date_format = 'dd/mm/yyyy';
ACCEPT datefrom DATE FORMAT 'dd/mm/yyyy' PROMPT 'Enter start date:'
ACCEPT dateto DATE FORMAT 'dd/mm/yyyy' PROMPT 'Enter to date:'

COLUMN productID FORMAT a10
COLUMN productName FORMAT a33

set linesize 120
set pagesize 60

TTITLE CENTER "LIST OF TOP 10 SELLING PRODUCTS BY QUANTITY SOLD" RIGHT 'Page:'
FORMAT 999 sql.pno SKIP 1
BREAK ON productID SKIP 1

SELECT * FROM (
SELECT p.productID, p.productName, p.category,
  SUM(oi.orderQuantity) AS quantitySold
  FROM product p
  JOIN ordered_item oi ON p.productID = oi.productID
  JOIN orders o ON oi.orderID = o.orderID
  WHERE o.orderDate BETWEEN '&datefrom' AND '&dateto'
  GROUP BY p.productID, p.productName, p.category
  ORDER BY quantitySold DESC
)
WHERE ROWNUM <= 10;

CLEAR COLUMNS
TTITLE OFF
```

3. Sample Output:



```
Session altered.

Enter start date:6/6/2024
Enter to date:7/7/2024
old   7:   WHERE o.orderDate BETWEEN '&datefrom' AND '&dateto'
new   7:   WHERE o.orderDate BETWEEN '6/6/2024' AND '7/7/2024'

                                    LIST OF TOP 10 SELLING PRODUCTS BY QUANTITY SOLD                           Page:    1
PRODUCTID  PRODUCTNAME                       CATEGORY                  QUANTITYSOLD
---------- --------------------------------- -------------------------- ------------
PD050      NVIDIA Quadro K6000               H-GPUs                          163880

PD082      NVIDIA Quadro 7000 6GB            H-GPUs                          103945

PD138      NVIDIA GeForce MX450 2GB          H-GPUs                           94337

PD066      NVIDIA Tesla V100 SXM2            S-Apps and Tools                 93022

PD107      NVIDIA GeForce MX130 4GB          H-Laptops and Workstations       92266

PD056      NVIDIA Shield TV 4K               H-Laptops and Workstations       91148

PD049      NVIDIA GeForce GT 710             S-Apps and Tools                 90628

PD135      NVIDIA Quadro RTX 8000 Max-Q      H-GPUs                           75156

PD068      NVIDIA GeForce GTX 760            S-Cloud Services                 72616

PD089      NVIDIA GeForce RTX 3080           S-Infrastructure                 67407
```

6.3 Query List of Pending Case (NG JIA HAO)

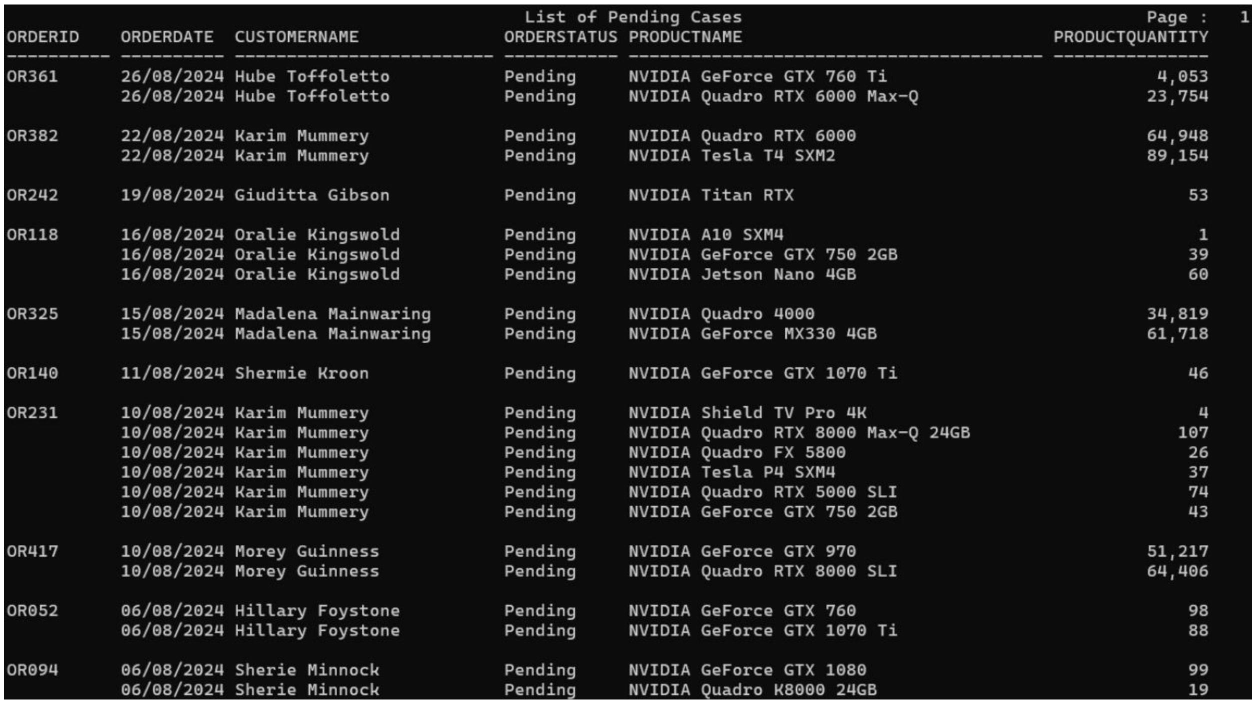
1. Purpose:

The status of pending cases plays a crucial role in ensuring the smooth operation of a business. It enables effective resource management, ensures customers are kept updated, facilitates financial management, and aids in future planning. Identifying pending orders empowers businesses to streamline their operations, enhance customer service, anticipate future requirements, and make informed decisions.

2. Query:

```
ALTER SESSION SET NLS_DATE_FORMAT = 'dd/mm/yyyy';
accept datefrom date format 'dd/mm/yyyy' prompt 'Enter datefrom :'
accept dateto date format 'dd/mm/yyyy' prompt 'Enter dateto :'

column orderID format A10
column customername format A25
column orderStatus format A11
column productQuantity format 9,999,990

set linesize 120
TTitle center 'List of Pending Cases' right 'Page :' format 999 SQL.PNO skip 1

BREAK ON REPORT ON orderID
COMPUTE SUM LABEL 'TotalUnit' of productQuantity ON REPORT

SELECT o.orderID, o.orderDate, c.name AS customerName, o.orderStatus, p.productName,
oi.orderQuantity AS productQuantity
FROM ORDERS o
JOIN customer c ON o.customerID = c.customerID
JOIN ORDERED_ITEM oi ON o.orderID = oi.orderID
JOIN PRODUCT p ON oi.productID = p.productID
WHERE o.orderStatus = 'Pending'
ORDER BY o.orderDate DESC,  o.orderID;

CLEAR COLUMNS
TTITLE OFF
```

3. Sample Output:



```
                                                  List of Pending Cases                                    Page :    1
ORDERID    ORDERDATE  CUSTOMERNAME              ORDERSTATUS PRODUCTNAME                              PRODUCTQUANTITY
---------- ---------- ------------------------- ----------- ---------------------------------------- ---------------
OR361      26/08/2024 Hube Toffoletto           Pending     NVIDIA GeForce GTX 760 Ti                          4,053
           26/08/2024 Hube Toffoletto           Pending     NVIDIA Quadro RTX 6000 Max-Q                      23,754

OR382      22/08/2024 Karim Mummery             Pending     NVIDIA Quadro RTX 6000                            64,948
           22/08/2024 Karim Mummery             Pending     NVIDIA Tesla T4 SXM2                              89,154

OR242      19/08/2024 Giuditta Gibson           Pending     NVIDIA Titan RTX                                      53

OR118      16/08/2024 Oralie Kingswold          Pending     NVIDIA A10 SXM4                                        1
           16/08/2024 Oralie Kingswold          Pending     NVIDIA GeForce GTX 750 2GB                            39
           16/08/2024 Oralie Kingswold          Pending     NVIDIA Jetson Nano 4GB                                60

OR325      15/08/2024 Madalena Mainwaring       Pending     NVIDIA Quadro 4000                                34,819
           15/08/2024 Madalena Mainwaring       Pending     NVIDIA GeForce MX330 4GB                          61,718

OR140      11/08/2024 Shermie Kroon             Pending     NVIDIA GeForce GTX 1070 Ti                            46

OR231      10/08/2024 Karim Mummery             Pending     NVIDIA Shield TV Pro 4K                                4
           10/08/2024 Karim Mummery             Pending     NVIDIA Quadro RTX 8000 Max-Q 24GB                    107
           10/08/2024 Karim Mummery             Pending     NVIDIA Quadro FX 5800                                 26
           10/08/2024 Karim Mummery             Pending     NVIDIA Tesla P4 SXM4                                  37
           10/08/2024 Karim Mummery             Pending     NVIDIA Quadro RTX 5000 SLI                            74
           10/08/2024 Karim Mummery             Pending     NVIDIA GeForce GTX 750 2GB                            43

OR417      10/08/2024 Morey Guinness            Pending     NVIDIA GeForce GTX 970                            51,217
           10/08/2024 Morey Guinness            Pending     NVIDIA Quadro RTX 8000 SLI                        64,406

OR052      06/08/2024 Hillary Foystone          Pending     NVIDIA GeForce GTX 760                                98
           06/08/2024 Hillary Foystone          Pending     NVIDIA GeForce GTX 1070 Ti                            88

OR094      06/08/2024 Sherie Minnock            Pending     NVIDIA GeForce GTX 1080                               99
           06/08/2024 Sherie Minnock            Pending     NVIDIA Quadro K8000 24GB                              19
```

6.4 Query Customer Total Spending (PECK JIA CHENG)

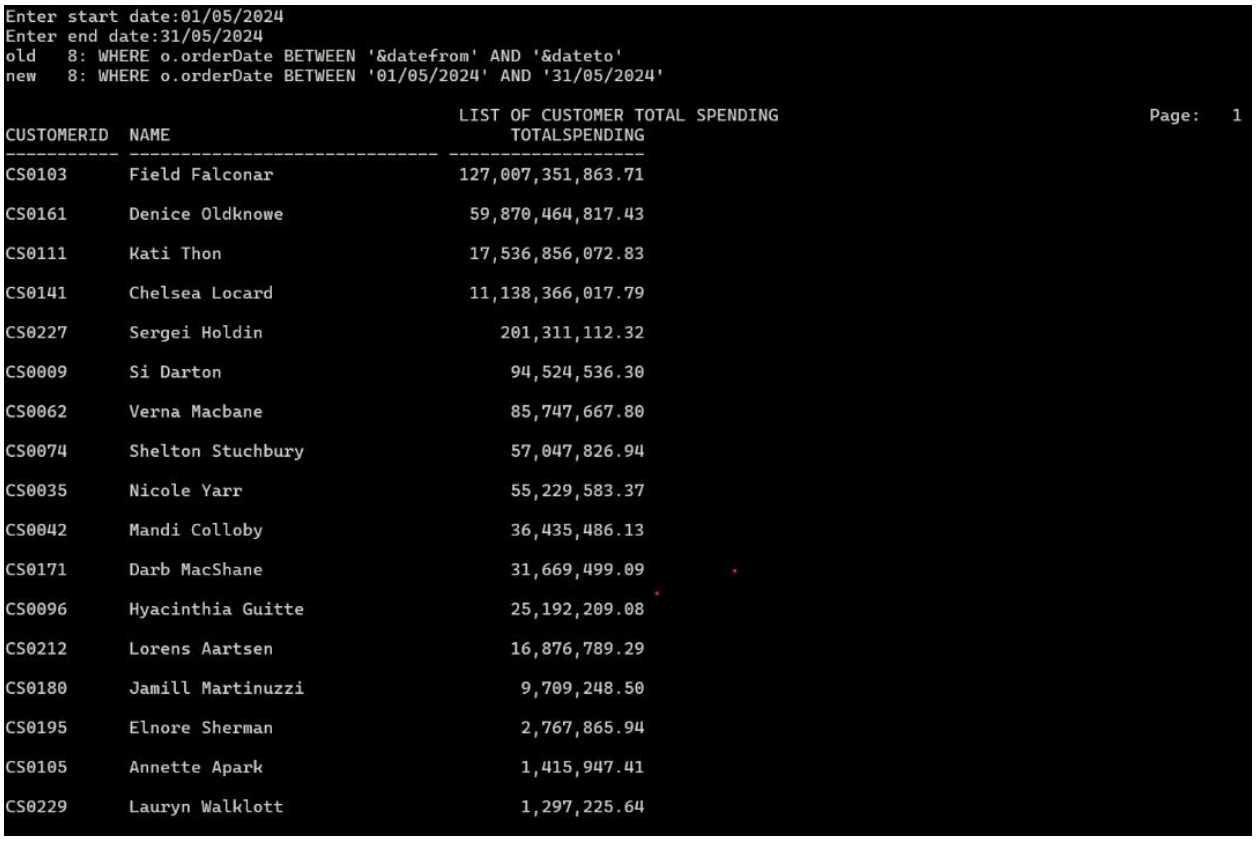
1. Purpose:

The purpose of calculating customer total spending is to gain insights into customer behavior and purchasing patterns. Furthermore, identify loyal customer who consistently make purchases, analyze customer lifetime value to assess the profitability of customer relationships. Also identify factors that influence customer spending and satisfaction. By understanding customer total spending, business can make data-driven decisions to improve customer satisfaction. Increase sales, and drive overall profitability.

2. Query:

```
ALTER SESSION SET NLS_DATE_FORMAT = 'dd/mm/yyyy';

ACCEPT datefrom DATE FORMAT 'dd/mm/yyyy' PROMPT 'Enter start date:'
ACCEPT dateto DATE FORMAT 'dd/mm/yyyy' PROMPT 'Enter end date:'

COLUMN "TotalSpending" FORMAT 999,999,999,990.99
COLUMN customerID FORMAT A11
COLUMN name FORMAT A30

SET LINESIZE 120
SET PAGESIZE 60
TTITLE CENTER "LIST OF CUSTOMER TOTAL SPENDING " RIGHT 'Page:' FORMAT 999 sql.pno
SKIP 1
BREAK ON customerID SKIP 1 ON name

SELECT
c.customerID,c.name,
SUM(oi.orderQuantity * p.sellingPrice) AS TotalSpending
FROM CUSTOMER C
JOIN orders o ON c.customerID = o.customerID
JOIN ordered_item oi ON o.orderID = oi.orderID
JOIN product p ON oi.productID = p.productID
WHERE o.orderDate BETWEEN '&datefrom' AND '&dateto'
GROUP BY c.customerID, c.name
ORDER BY TotalSpending DESC;

CLEAR COLUMNS
TTITLE OFF
```

3. Sample Output:



```
Enter start date:01/05/2024
Enter end date:31/05/2024
old   8: WHERE o.orderDate BETWEEN '&datefrom' AND '&dateto'
new   8: WHERE o.orderDate BETWEEN '01/05/2024' AND '31/05/2024'

                                            LIST OF CUSTOMER TOTAL SPENDING                                    Page:    1
CUSTOMERID  NAME                                TOTALSPENDING
----------- ------------------------------ -------------------
CS0103      Field Falconar                  127,007,351,863.71

CS0161      Denice Oldknowe                  59,870,464,817.43

CS0111      Kati Thon                        17,536,856,072.83

CS0141      Chelsea Locard                   11,138,366,017.79

CS0227      Sergei Holdin                       201,311,112.32

CS0009      Si Darton                            94,524,536.30

CS0062      Verna Macbane                        85,747,667.80

CS0074      Shelton Stuchbury                    57,047,826.94

CS0035      Nicole Yarr                          55,229,583.37

CS0042      Mandi Colloby                        36,435,486.13

CS0171      Darb MacShane                        31,669,499.09

CS0096      Hyacinthia Guitte                    25,192,209.08

CS0212      Lorens Aartsen                       16,876,789.29

CS0180      Jamill Martinuzzi                     9,709,248.50

CS0195      Elnore Sherman                        2,767,865.94

CS0105      Annette Apark                         1,415,947.41

CS0229      Lauryn Walklott                       1,297,225.64
```

6.5 Query Supplier Contribution (LOW KAI QIN)

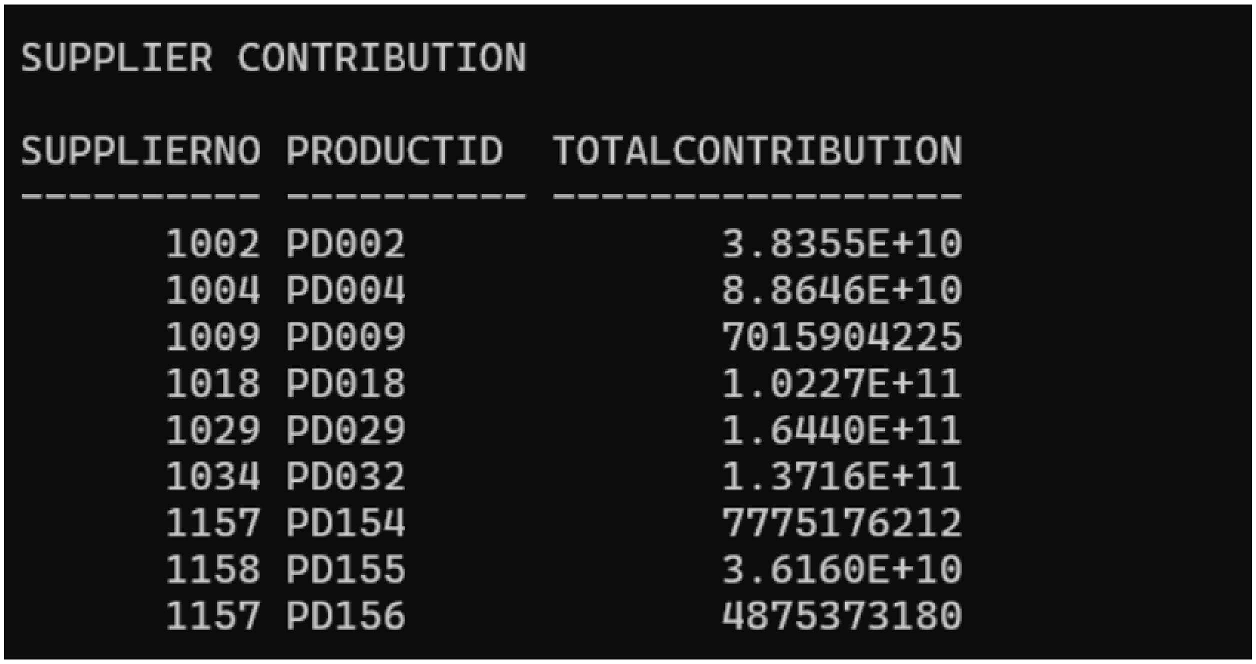
1. **Purpose:**

The purpose of tracking supplier contribution is to evaluate the role each supplier plays in the fulfillment of customer orders, ensuring that products are available when needed, at the right cost, and with the required quality. This helps in managing supplier relationships, optimizing supply chain processes, and reducing operational risks.

2. **Query:**

```
TTITLE left'SUPPLIER CONTRIBUTION' SKIP 2;
SET LINESIZE 120 CLEAR COLUMNS
SELECT
        SUPPLIER_PRODUCT.supplierNo,
        SUPPLIER_PRODUCT.productID,
        SUM(ORDERED_ITEM.orderQuantity * SUPPLIER_PRODUCT.supplyPrice) AS
        totalContribution FROM
SUPPLIER_PRODUCT
JOIN
        ORDERED_ITEM ON SUPPLIER_PRODUCT.productID = ORDERED_ITEM.productID
GROUP BY
        SUPPLIER_PRODUCT.supplierNo, SUPPLIER_PRODUCT.productID;
TTITLE OFF
CLEAR COLUMNS;
```

3. **Sample Output:**



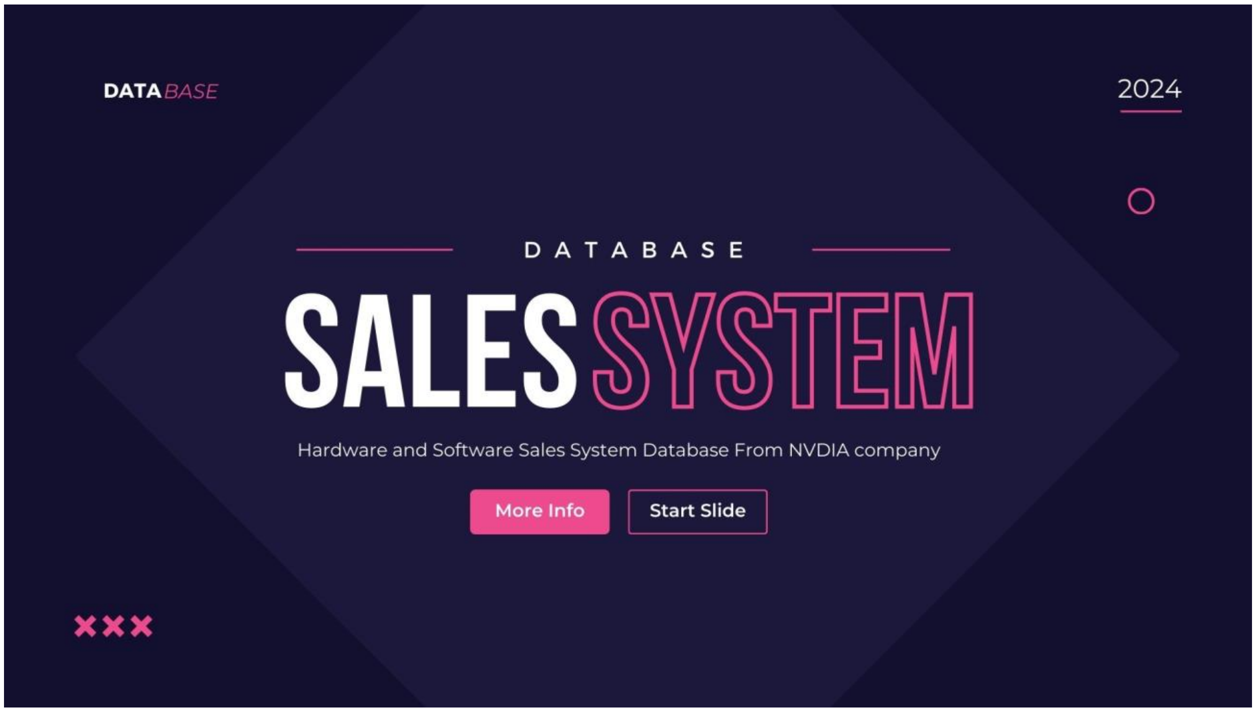
```
SUPPLIER CONTRIBUTION

SUPPLIERNO PRODUCTID  TOTALCONTRIBUTION
---------- ---------- -----------------
      1002 PD002             3.8355E+10
      1004 PD004             8.8646E+10
      1009 PD009             7015904225
      1018 PD018             1.0227E+11
      1029 PD029             1.6440E+11
      1034 PD032             1.3716E+11
      1157 PD154             7775176212
      1158 PD155             3.6160E+10
      1157 PD156             4875373180
```

REFERENCES

Mockaroo. (n.d.). Retrieved from Mockaroo: https://www.mockaroo.com/mock_apis

NVIDIA. (n.d.). Retrieved from NVDIA: https://www.nvidia.com/en-us/

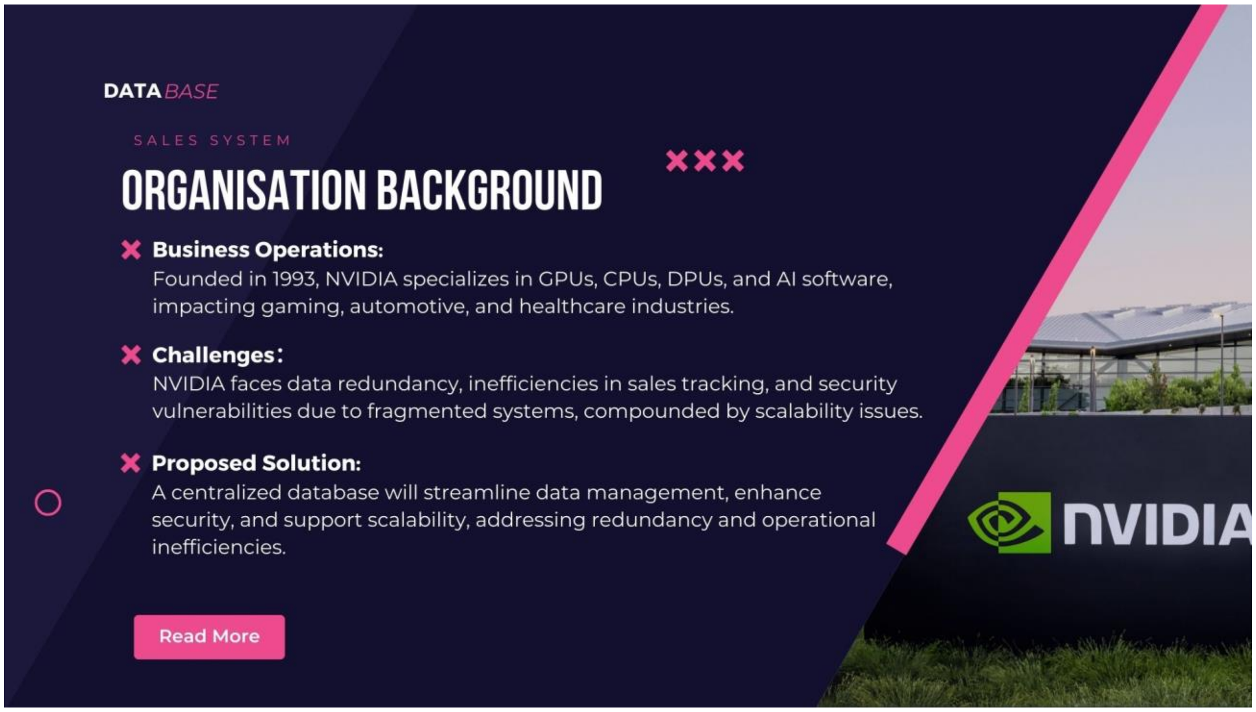
APPENDIX


DATA BASE
2024
More Info
Start Slide

DATABASE

SALES SYSTEM

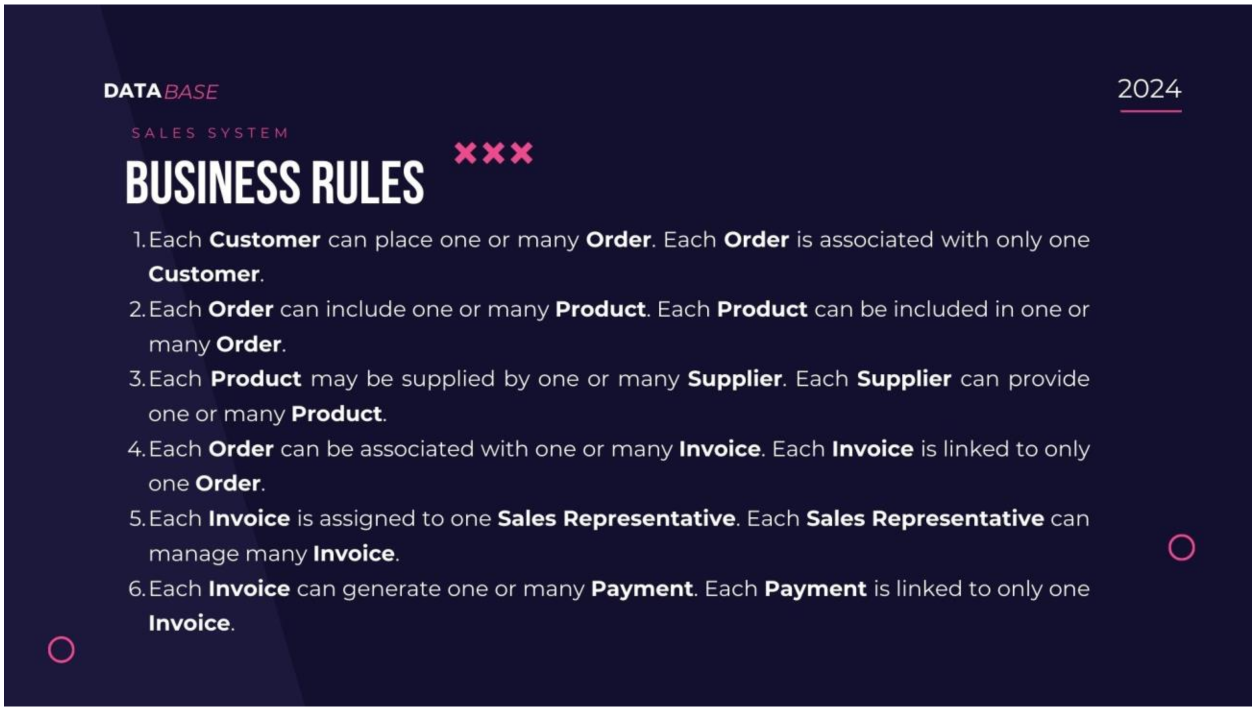
Hardware and Software Sales System Database From NVDIA company


DATA BASE
Read More
NVIDIA

SALES SYSTEM

ORGANISATION BACKGROUND

- **Business Operations:** Founded in 1993, NVIDIA specializes in GPUs, CPUs, DPUs, and AI software, impacting gaming, automotive, and healthcare industries.
- **Challenges:** NVIDIA faces data redundancy, inefficiencies in sales tracking, and security vulnerabilities due to fragmented systems, compounded by scalability issues.
- **Proposed Solution:** A centralized database will streamline data management, enhance security, and support scalability, addressing redundancy and operational inefficiencies.



DATA*BASE*

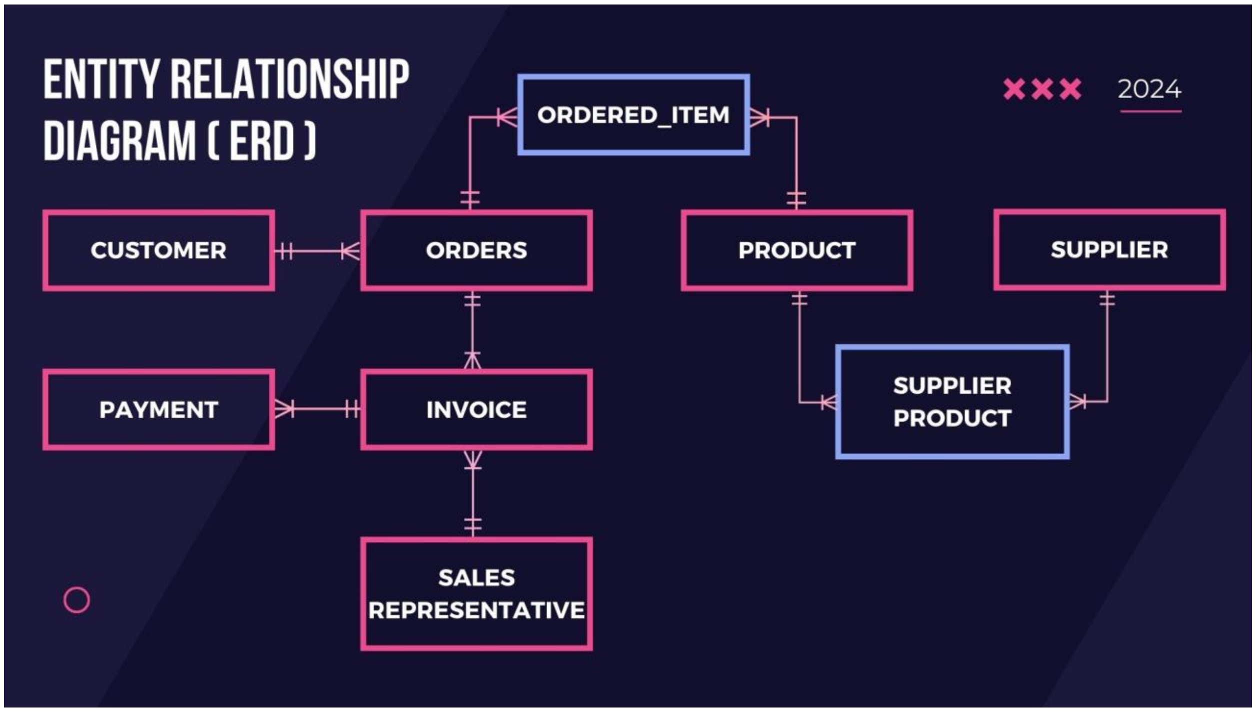
2024

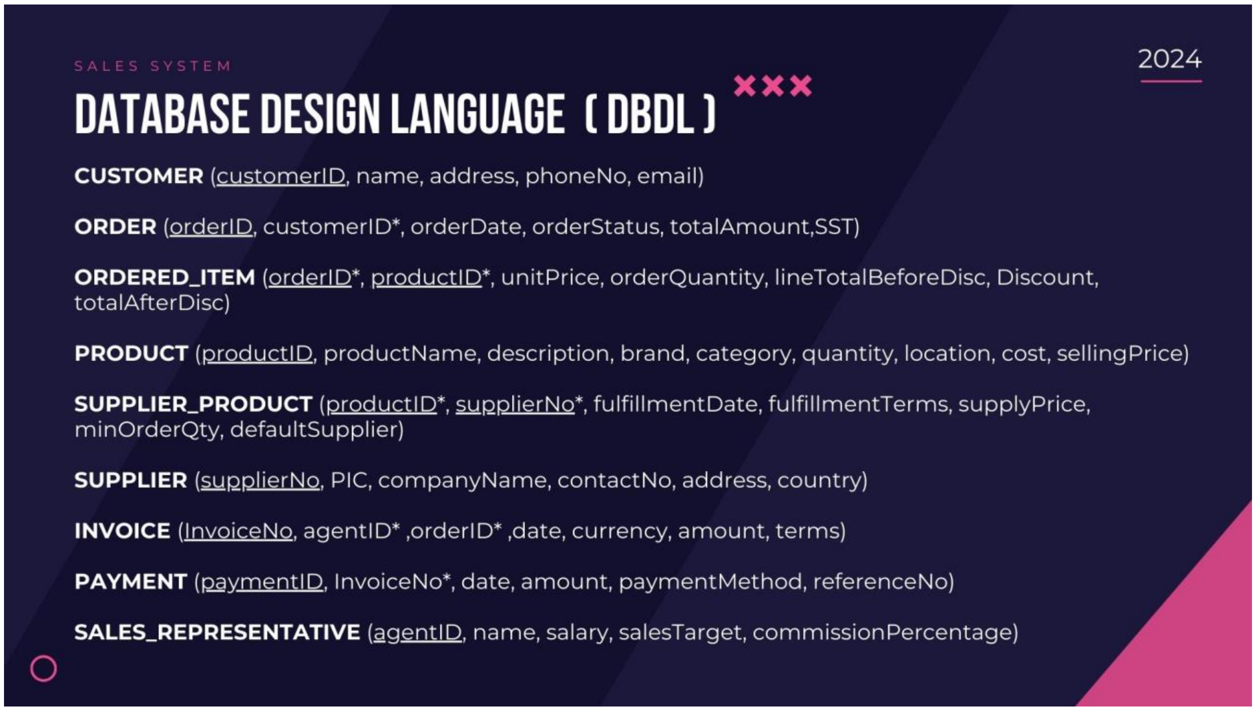
SALES SYSTEM

BUSINESS RULES

1. Each **Customer** can place one or many **Order**. Each **Order** is associated with only one **Customer**.
2. Each **Order** can include one or many **Product**. Each **Product** can be included in one or many **Order**.
3. Each **Product** may be supplied by one or many **Supplier**. Each **Supplier** can provide one or many **Product**.
4. Each **Order** can be associated with one or many **Invoice**. Each **Invoice** is linked to only one **Order**.
5. Each **Invoice** is assigned to one **Sales Representative**. Each **Sales Representative** can manage many **Invoice**.
6. Each **Invoice** can generate one or many **Payment**. Each **Payment** is linked to only one **Invoice**.

ENTITY RELATIONSHIP DIAGRAM (ERD)





SALES SYSTEM

2024

DATABASE DESIGN LANGUAGE (DBDL)

CUSTOMER (customerID, name, address, phoneNo, email)

ORDER (orderID, customerID*, orderDate, orderStatus, totalAmount,SST)

ORDERED_ITEM (orderID*, productID*, unitPrice, orderQuantity, lineTotalBeforeDisc, Discount, totalAfterDisc)

PRODUCT (productID, productName, description, brand, category, quantity, location, cost, sellingPrice)

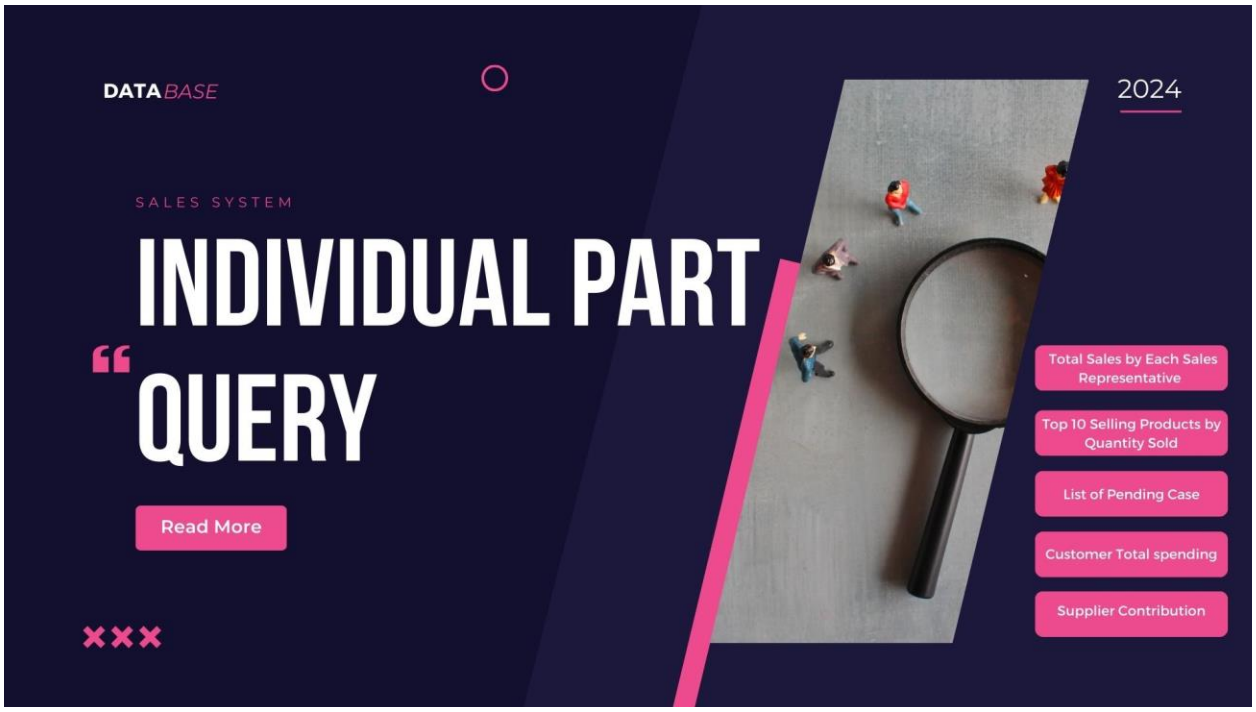
SUPPLIER_PRODUCT (productID*, supplierNo*, fulfillmentDate, fulfillmentTerms, supplyPrice, minOrderQty, defaultSupplier)

SUPPLIER (supplierNo, PIC, companyName, contactNo, address, country)

INVOICE (InvoiceNo, agentID* ,orderID* ,date, currency, amount, terms)

PAYMENT (paymentID, InvoiceNo*, date, amount, paymentMethod, referenceNo)

SALES_REPRESENTATIVE (agentID, name, salary, salesTarget, commissionPercentage)



DATA*BASE*

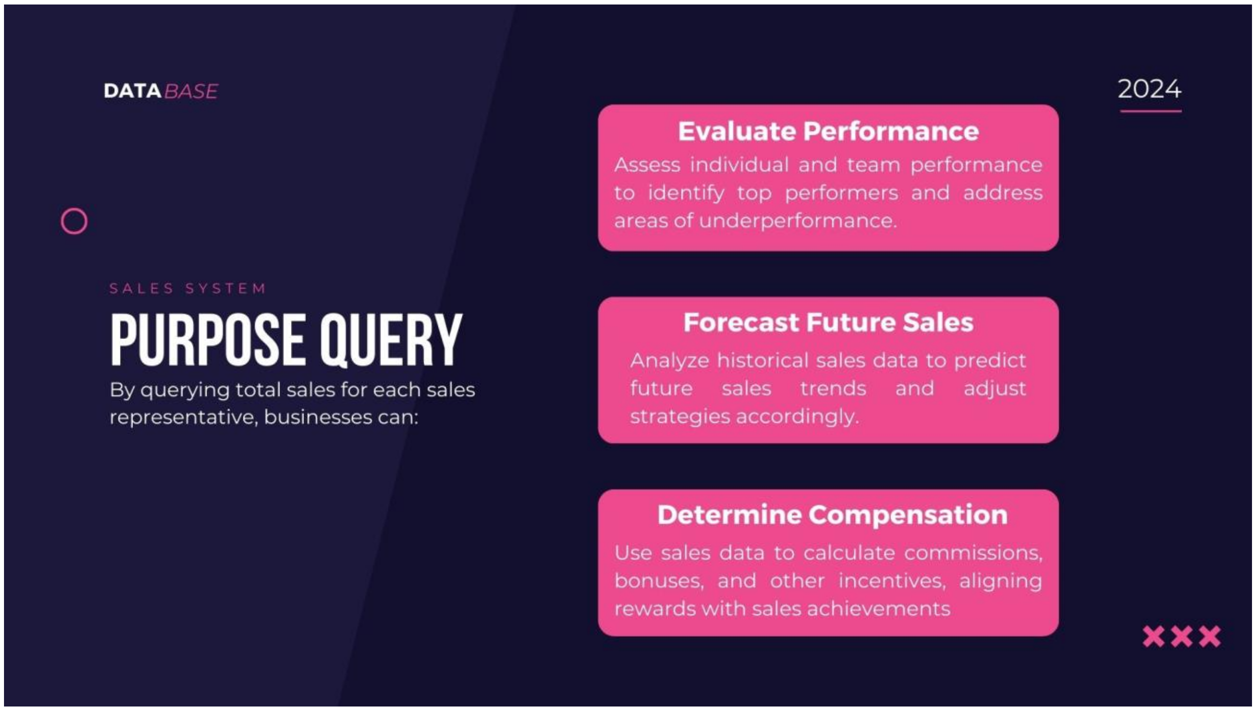
2024

SALES SYSTEM

“INDIVIDUAL PART QUERY

Read More

- Total Sales by Each Sales Representative
- Top 10 Selling Products by Quantity Sold
- List of Pending Case
- Customer Total spending
- Supplier Contribution



DATA*BASE*

2024

SALES SYSTEM

PURPOSE QUERY

By querying total sales for each sales representative, businesses can:

Evaluate Performance

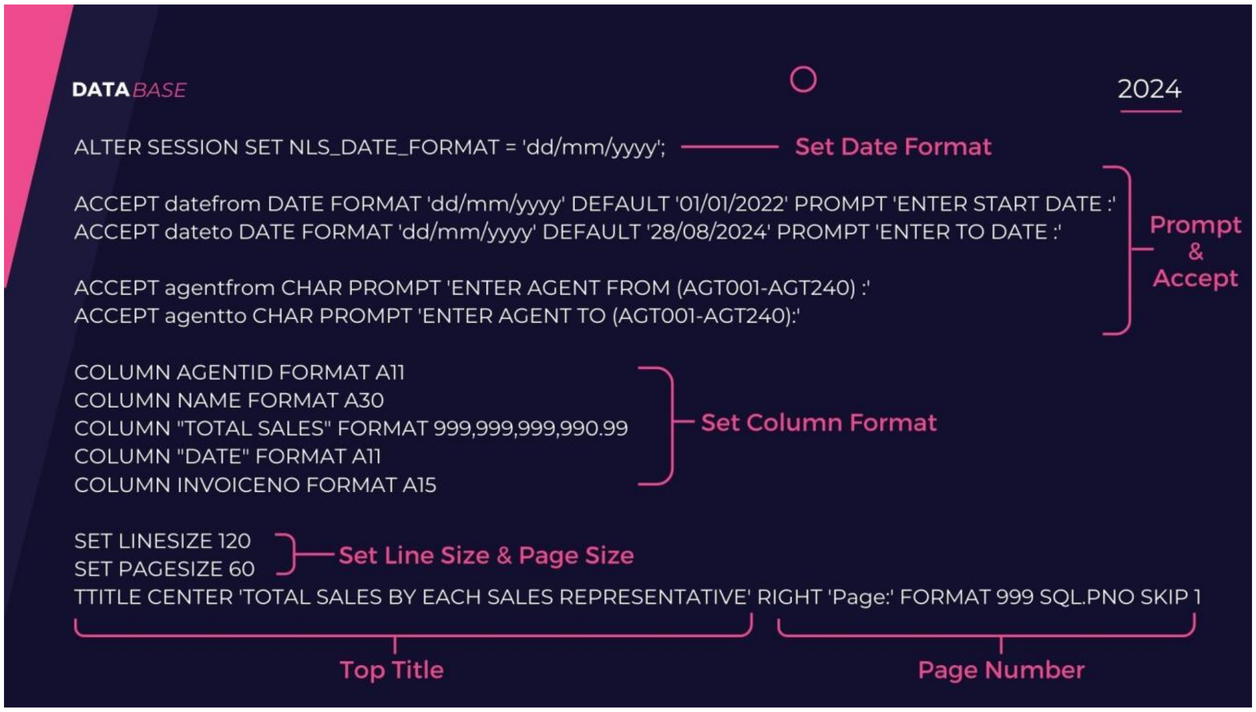
Assess individual and team performance to identify top performers and address areas of underperformance.

Forecast Future Sales

Analyze historical sales data to predict future sales trends and adjust strategies accordingly.

Determine Compensation

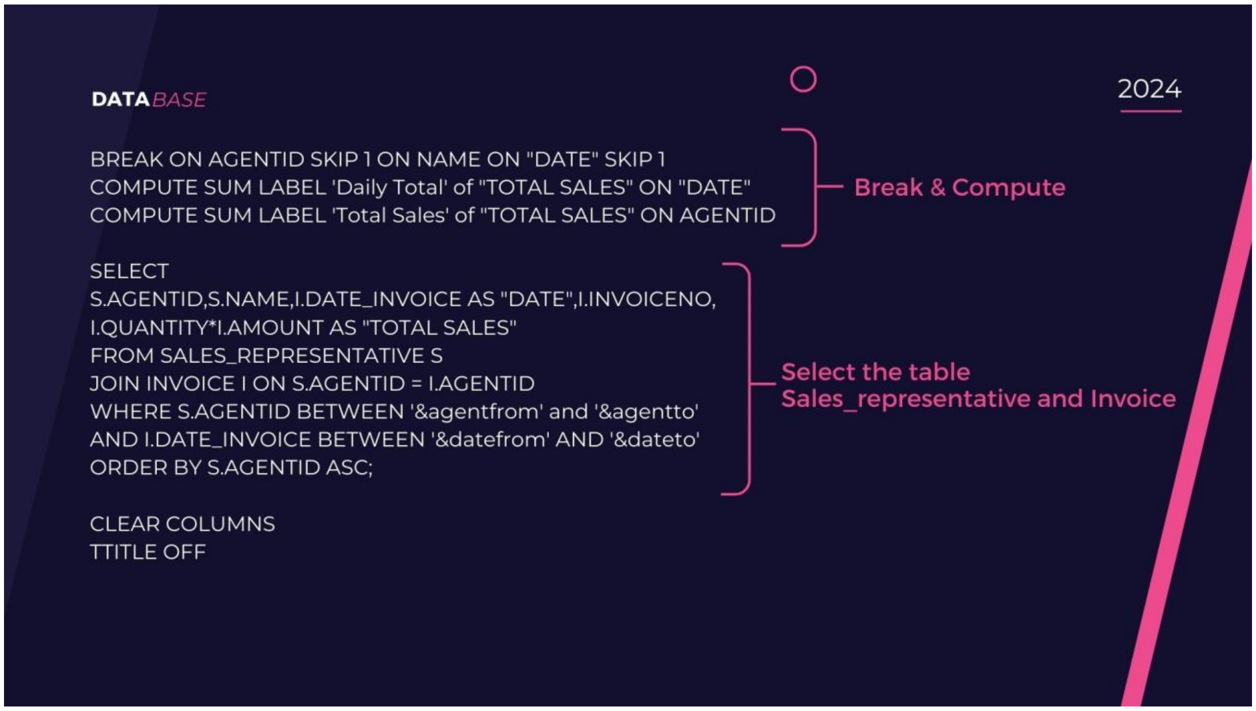
Use sales data to calculate commissions, bonuses, and other incentives, aligning rewards with sales achievements



DATA*BASE*

2024

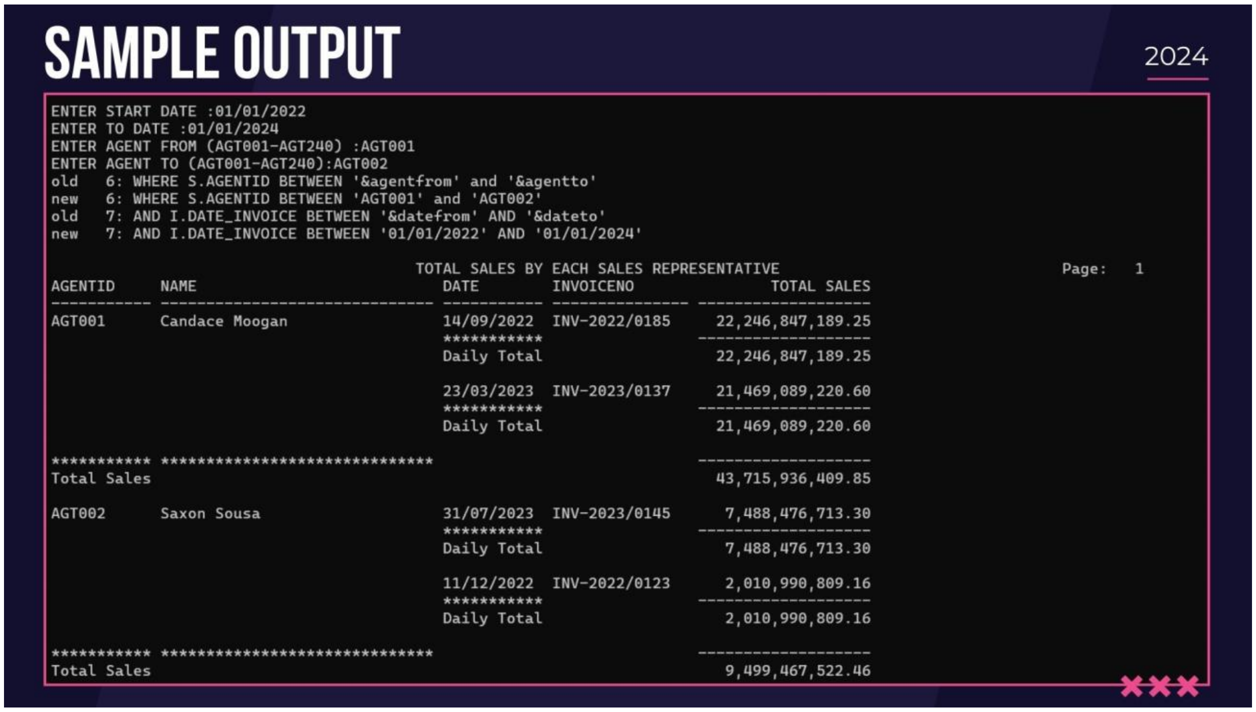
```
ALTER SESSION SET NLS_DATE_FORMAT = 'dd/mm/yyyy';

ACCEPT datefrom DATE FORMAT 'dd/mm/yyyy' DEFAULT '01/01/2022' PROMPT 'ENTER START DATE :'
ACCEPT dateto DATE FORMAT 'dd/mm/yyyy' DEFAULT '28/08/2024' PROMPT 'ENTER TO DATE :'

ACCEPT agentfrom CHAR PROMPT 'ENTER AGENT FROM (AGT001-AGT240) :'
ACCEPT agentto CHAR PROMPT 'ENTER AGENT TO (AGT001-AGT240):'

COLUMN AGENTID FORMAT A11
COLUMN NAME FORMAT A30
COLUMN "TOTAL SALES" FORMAT 999,999,999,990.99
COLUMN "DATE" FORMAT A11
COLUMN INVOICENO FORMAT A15

SET LINESIZE 120
SET PAGESIZE 60
TTITLE CENTER 'TOTAL SALES BY EACH SALES REPRESENTATIVE' RIGHT 'Page:' FORMAT 999 SQL.PNO SKIP 1
```



DATA*BASE*

2024

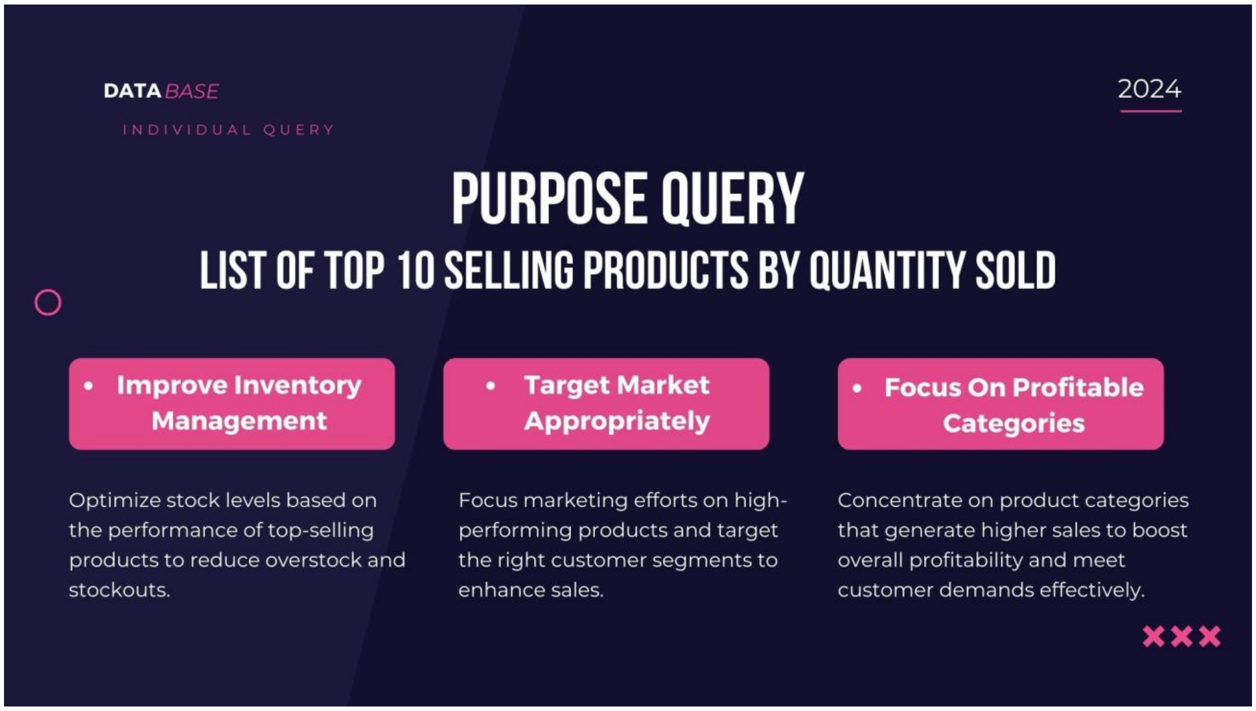
```
BREAK ON AGENTID SKIP 1 ON NAME ON "DATE" SKIP 1
COMPUTE SUM LABEL 'Daily Total' of "TOTAL SALES" ON "DATE"
COMPUTE SUM LABEL 'Total Sales' of "TOTAL SALES" ON AGENTID

SELECT
S.AGENTID,S.NAME,I.DATE_INVOICE AS "DATE",I.INVOICENO,
I.QUANTITY*I.AMOUNT AS "TOTAL SALES"
FROM SALES_REPRESENTATIVE S
JOIN INVOICE I ON S.AGENTID = I.AGENTID
WHERE S.AGENTID BETWEEN '&agentfrom' and '&agentto'
AND I.DATE_INVOICE BETWEEN '&datefrom' AND '&dateto'
ORDER BY S.AGENTID ASC;

CLEAR COLUMNS
TTITLE OFF
```



SAMPLE OUTPUT

2024

```
ENTER START DATE :01/01/2022
ENTER TO DATE :01/01/2024
ENTER AGENT FROM (AGT001-AGT240) :AGT001
ENTER AGENT TO (AGT001-AGT240):AGT002
old   6: WHERE S.AGENTID BETWEEN '&agentfrom' and '&agentto'
new   6: WHERE S.AGENTID BETWEEN 'AGT001' and 'AGT002'
old   7: AND I.DATE_INVOICE BETWEEN '&datefrom' AND '&dateto'
new   7: AND I.DATE_INVOICE BETWEEN '01/01/2022' AND '01/01/2024'

                                        TOTAL SALES BY EACH SALES REPRESENTATIVE                               Page:    1
AGENTID     NAME                           DATE        INVOICENO               TOTAL SALES
----------- ------------------------------ ----------- ---------------- --------------------
AGT001      Candace Moogan                 14/09/2022  INV-2022/0185       22,246,847,189.25
                                           ***********                  --------------------
                                           Daily Total                     22,246,847,189.25

                                           23/03/2023  INV-2023/0137       21,469,089,220.60
                                           ***********                  --------------------
                                           Daily Total                     21,469,089,220.60

*********** ******************************                              --------------------
Total Sales                                                                43,715,936,409.85

AGT002      Saxon Sousa                    31/07/2023  INV-2023/0145        7,488,476,713.30
                                           ***********                  --------------------
                                           Daily Total                      7,488,476,713.30

                                           11/12/2022  INV-2022/0123        2,010,990,809.16
                                           ***********                  --------------------
                                           Daily Total                      2,010,990,809.16

*********** ******************************                              --------------------
Total Sales                                                                 9,499,467,522.46
```



DATA BASE

INDIVIDUAL QUERY

2024

PURPOSE QUERY

LIST OF TOP 10 SELLING PRODUCTS BY QUANTITY SOLD

- **Improve Inventory Management**

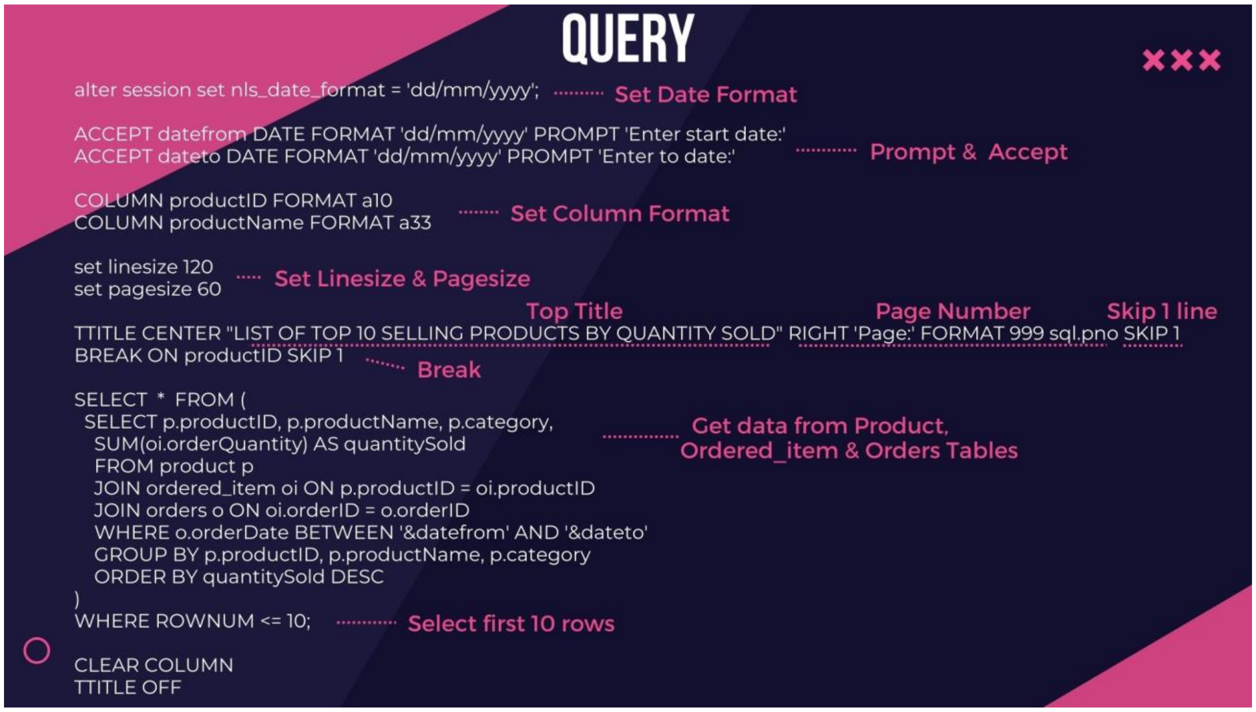
Optimize stock levels based on the performance of top-selling products to reduce overstock and stockouts.

- **Target Market Appropriately**

Focus marketing efforts on high-performing products and target the right customer segments to enhance sales.

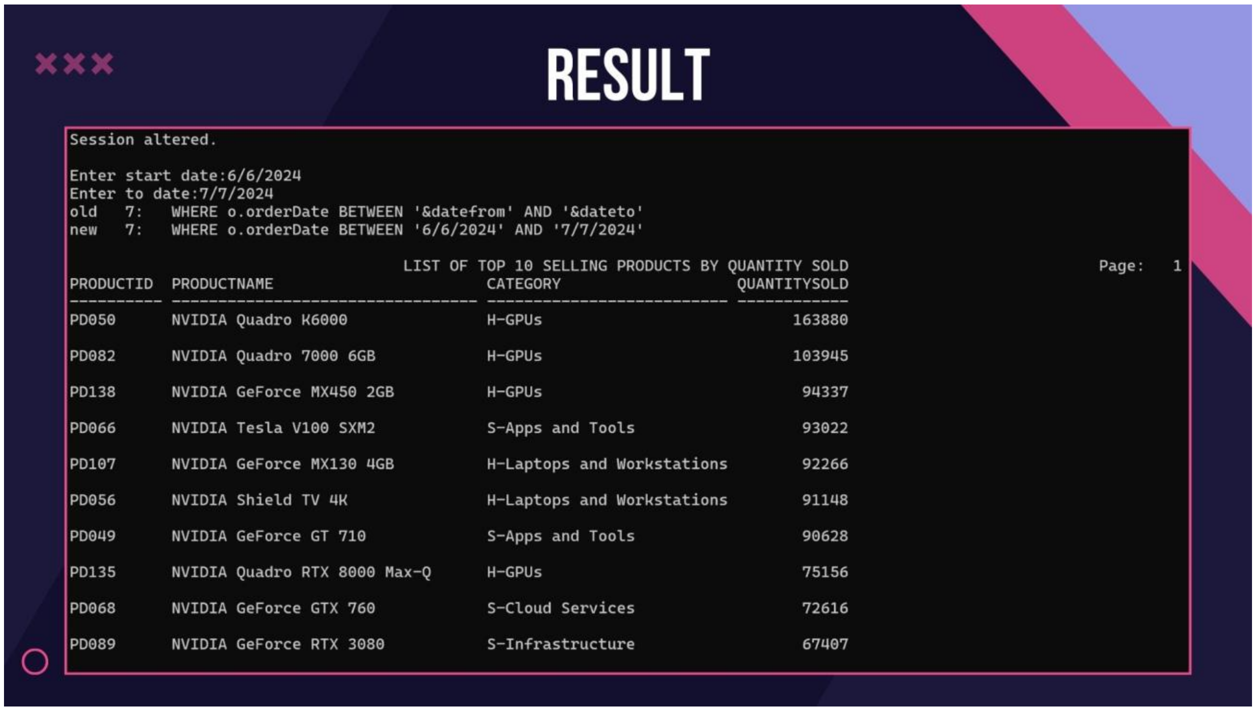
- **Focus On Profitable Categories**

Concentrate on product categories that generate higher sales to boost overall profitability and meet customer demands effectively.



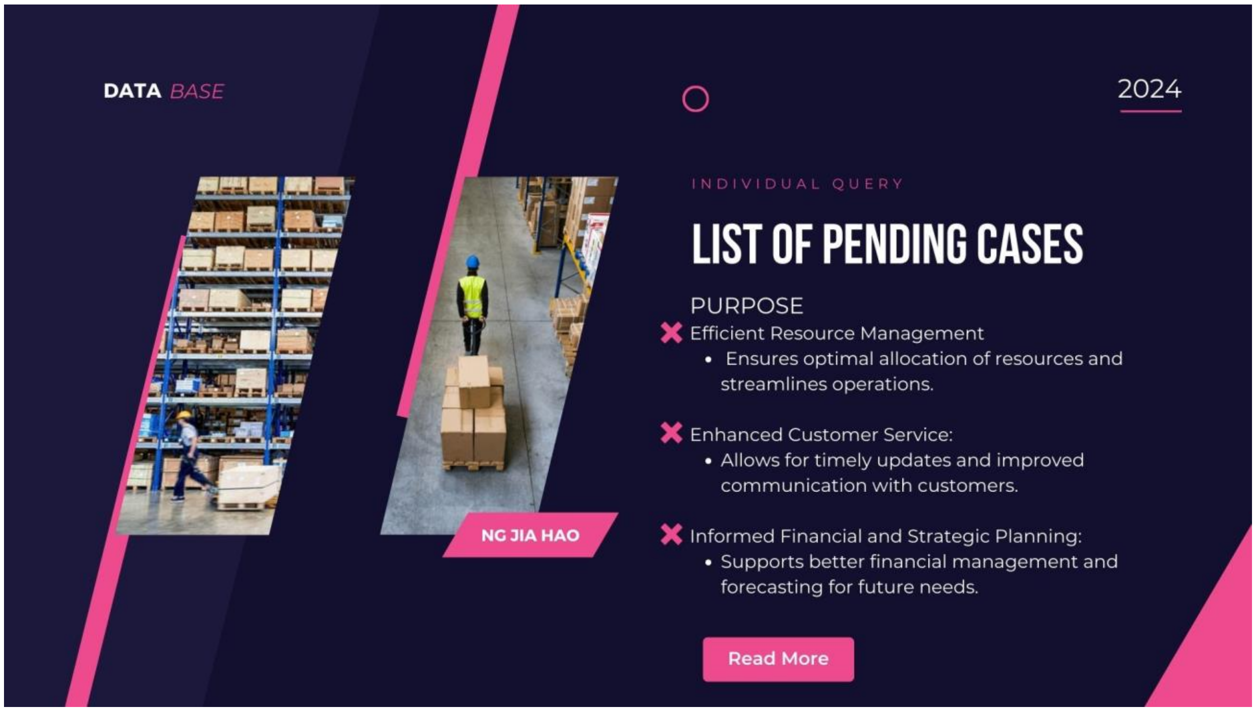
QUERY

```
alter session set nls_date_format = 'dd/mm/yyyy';

ACCEPT datefrom DATE FORMAT 'dd/mm/yyyy' PROMPT 'Enter start date:'
ACCEPT dateto DATE FORMAT 'dd/mm/yyyy' PROMPT 'Enter to date:'

COLUMN productID FORMAT a10
COLUMN productName FORMAT a33

set linesize 120
set pagesize 60

TTITLE CENTER "LIST OF TOP 10 SELLING PRODUCTS BY QUANTITY SOLD" RIGHT 'Page:' FORMAT 999 sql.pno SKIP 1
BREAK ON productID SKIP 1

SELECT * FROM (
 SELECT p.productID, p.productName, p.category,
  SUM(oi.orderQuantity) AS quantitySold
  FROM product p
  JOIN ordered_item oi ON p.productID = oi.productID
  JOIN orders o ON oi.orderID = o.orderID
  WHERE o.orderDate BETWEEN '&datefrom' AND '&dateto'
  GROUP BY p.productID, p.productName, p.category
  ORDER BY quantitySold DESC
)
WHERE ROWNUM <= 10;

CLEAR COLUMN
TTITLE OFF
```



RESULT

```
Session altered.

Enter start date:6/6/2024
Enter to date:7/7/2024
old   7:   WHERE o.orderDate BETWEEN '&datefrom' AND '&dateto'
new   7:   WHERE o.orderDate BETWEEN '6/6/2024' AND '7/7/2024'

                                    LIST OF TOP 10 SELLING PRODUCTS BY QUANTITY SOLD                           Page:    1
PRODUCTID  PRODUCTNAME                        CATEGORY                        QUANTITYSOLD
---------- ---------------------------------- -------------------------- ------------
PD050      NVIDIA Quadro K6000                H-GPUs                           163880

PD082      NVIDIA Quadro 7000 6GB             H-GPUs                           103945

PD138      NVIDIA GeForce MX450 2GB           H-GPUs                            94337

PD066      NVIDIA Tesla V100 SXM2             S-Apps and Tools                  93022

PD107      NVIDIA GeForce MX130 4GB           H-Laptops and Workstations        92266

PD056      NVIDIA Shield TV 4K                H-Laptops and Workstations        91148

PD049      NVIDIA GeForce GT 710              S-Apps and Tools                  90628

PD135      NVIDIA Quadro RTX 8000 Max-Q       H-GPUs                            75156

PD068      NVIDIA GeForce GTX 760             S-Cloud Services                  72616

PD089      NVIDIA GeForce RTX 3080            S-Infrastructure                  67407
```



DATA *BASE*

2024

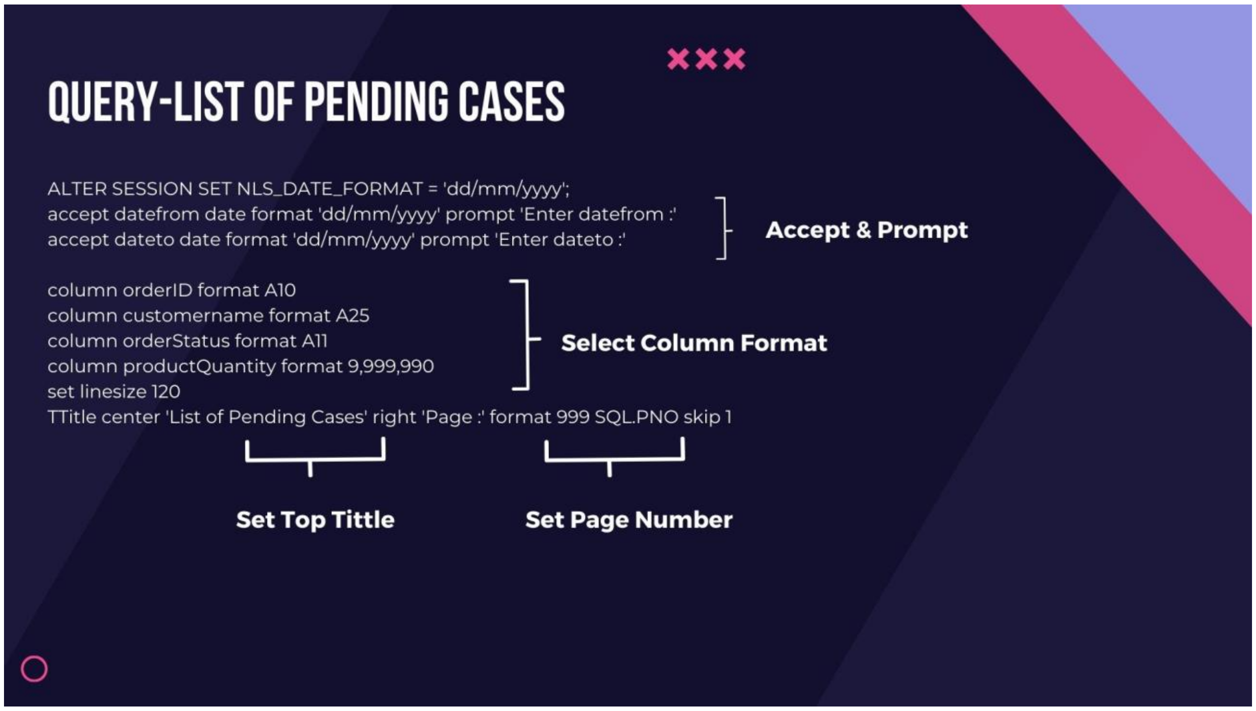
INDIVIDUAL QUERY

LIST OF PENDING CASES

NG JIA HAO

PURPOSE

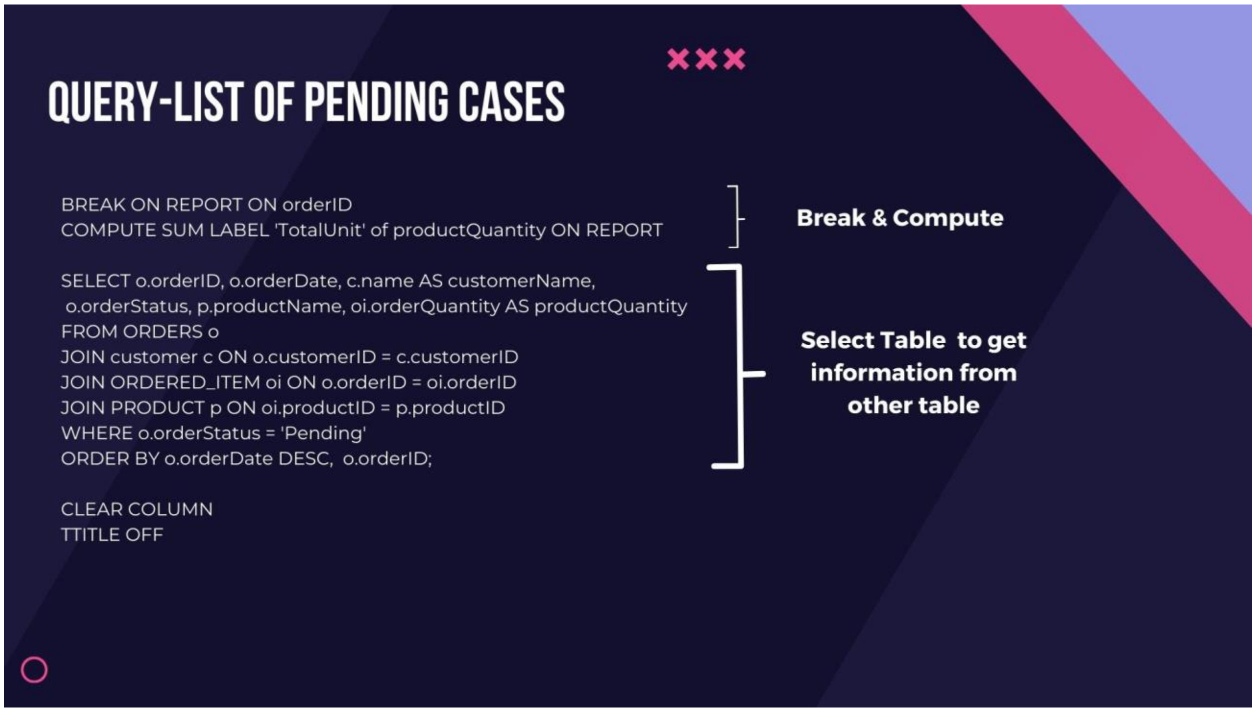
- Efficient Resource Management
  - Ensures optimal allocation of resources and streamlines operations.
- Enhanced Customer Service:
  - Allows for timely updates and improved communication with customers.
- Informed Financial and Strategic Planning:
  - Supports better financial management and forecasting for future needs.



QUERY-LIST OF PENDING CASES

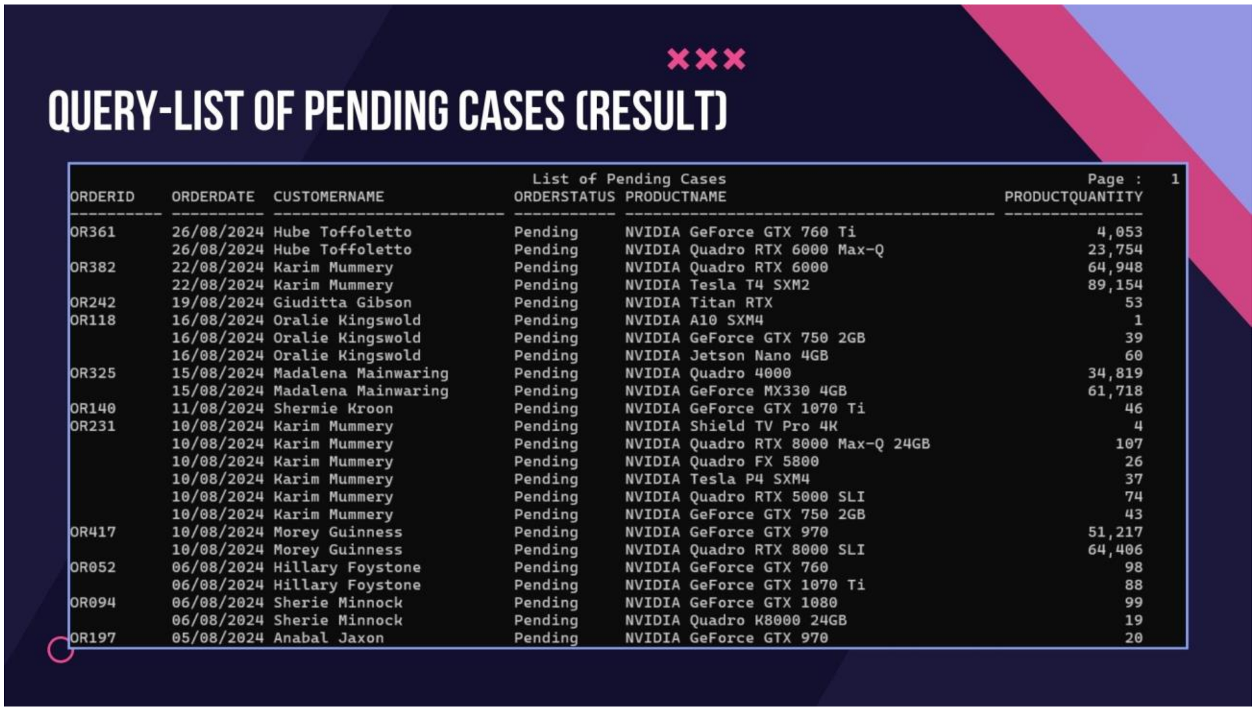
```
ALTER SESSION SET NLS_DATE_FORMAT = 'dd/mm/yyyy';
accept datefrom date format 'dd/mm/yyyy' prompt 'Enter datefrom :'
accept dateto date format 'dd/mm/yyyy' prompt 'Enter dateto :'

column orderID format A10
column customername format A25
column orderStatus format A11
column productQuantity format 9,999,990
set linesize 120
TTitle center 'List of Pending Cases' right 'Page :' format 999 SQL.PNO skip 1
```

Accept & Prompt
Select Column Format
Set Top Tittle
Set Page Number



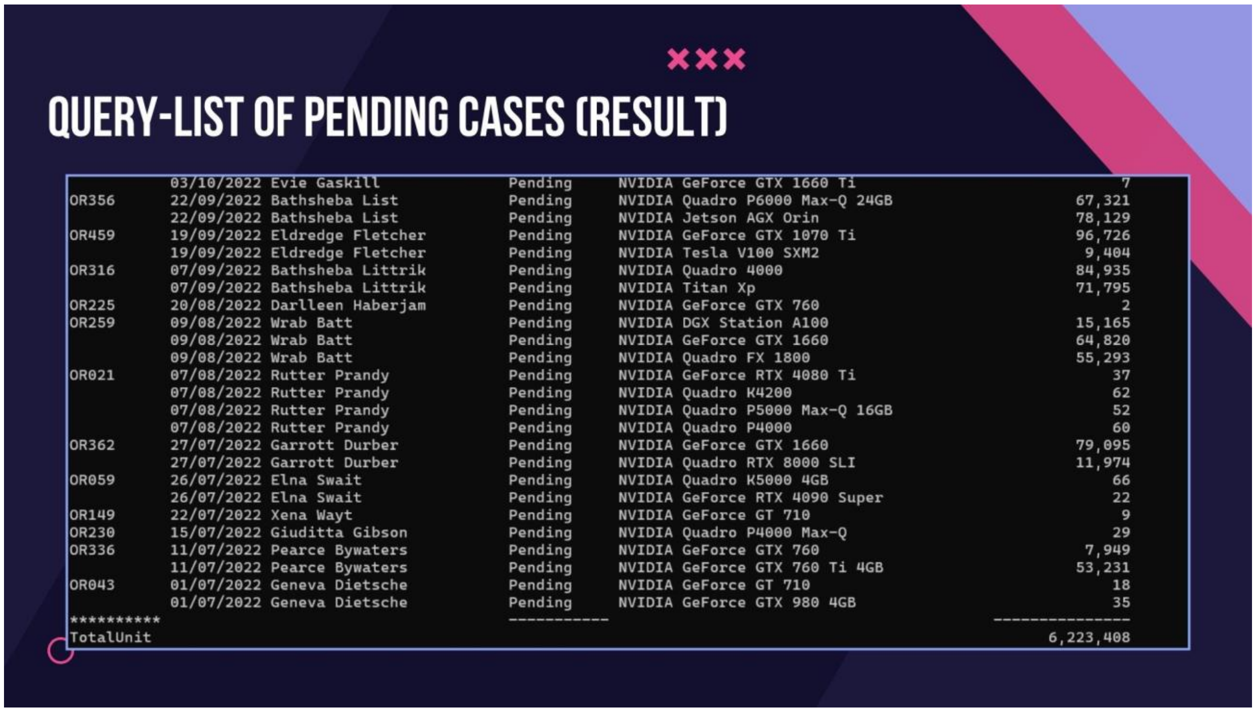
QUERY-LIST OF PENDING CASES

```
BREAK ON REPORT ON orderID
COMPUTE SUM LABEL 'TotalUnit' of productQuantity ON REPORT

SELECT o.orderID, o.orderDate, c.name AS customerName,
 o.orderStatus, p.productName, oi.orderQuantity AS productQuantity
FROM ORDERS o
JOIN customer c ON o.customerID = c.customerID
JOIN ORDERED_ITEM oi ON o.orderID = oi.orderID
JOIN PRODUCT p ON oi.productID = p.productID
WHERE o.orderStatus = 'Pending'
ORDER BY o.orderDate DESC,  o.orderID;

CLEAR COLUMN
TTITLE OFF
```



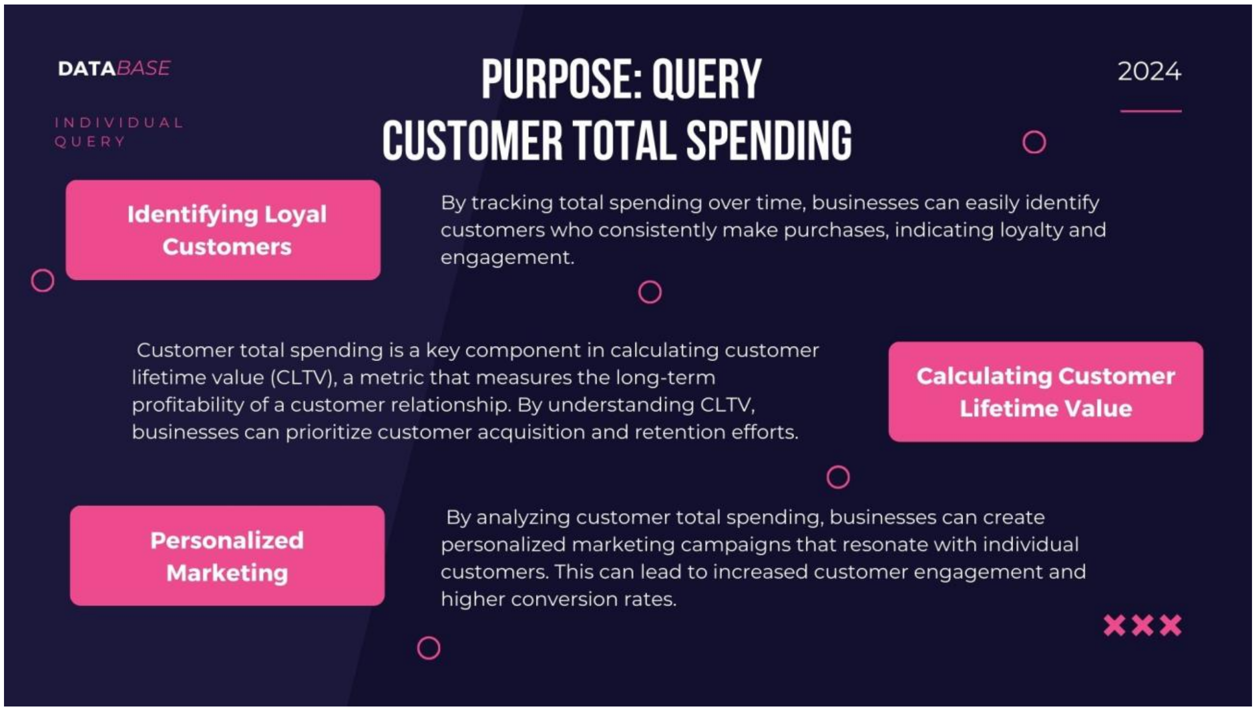
QUERY-LIST OF PENDING CASES (RESULT)

```
                                               List of Pending Cases                                             Page :    1
ORDERID    ORDERDATE  CUSTOMERNAME              ORDERSTATUS PRODUCTNAME                                PRODUCTQUANTITY
---------- ---------- ------------------------- ----------- ---------------------------------------- ---------------
OR361      26/08/2024 Hube Toffoletto           Pending     NVIDIA GeForce GTX 760 Ti                           4,053
           26/08/2024 Hube Toffoletto           Pending     NVIDIA Quadro RTX 6000 Max-Q                       23,754
OR382      22/08/2024 Karim Mummery             Pending     NVIDIA Quadro RTX 6000                             64,948
           22/08/2024 Karim Mummery             Pending     NVIDIA Tesla T4 SXM2                               89,154
OR242      19/08/2024 Giuditta Gibson           Pending     NVIDIA Titan RTX                                       53
OR118      16/08/2024 Oralie Kingswold          Pending     NVIDIA A10 SXM4                                         1
           16/08/2024 Oralie Kingswold          Pending     NVIDIA GeForce GTX 750 2GB                             39
           16/08/2024 Oralie Kingswold          Pending     NVIDIA Jetson Nano 4GB                                 60
OR325      15/08/2024 Madalena Mainwaring       Pending     NVIDIA Quadro 4000                                 34,819
           15/08/2024 Madalena Mainwaring       Pending     NVIDIA GeForce MX330 4GB                           61,718
OR140      11/08/2024 Shermie Kroon             Pending     NVIDIA GeForce GTX 1070 Ti                             46
OR231      10/08/2024 Karim Mummery             Pending     NVIDIA Shield TV Pro 4K                                 4
           10/08/2024 Karim Mummery             Pending     NVIDIA Quadro RTX 8000 Max-Q 24GB                     107
           10/08/2024 Karim Mummery             Pending     NVIDIA Quadro FX 5800                                  26
           10/08/2024 Karim Mummery             Pending     NVIDIA Tesla P4 SXM4                                   37
           10/08/2024 Karim Mummery             Pending     NVIDIA Quadro RTX 5000 SLI                             74
           10/08/2024 Karim Mummery             Pending     NVIDIA GeForce GTX 750 2GB                             43
OR417      10/08/2024 Morey Guinness            Pending     NVIDIA GeForce GTX 970                             51,217
           10/08/2024 Morey Guinness            Pending     NVIDIA Quadro RTX 8000 SLI                         64,406
OR052      06/08/2024 Hillary Foystone          Pending     NVIDIA GeForce GTX 760                                 98
           06/08/2024 Hillary Foystone          Pending     NVIDIA GeForce GTX 1070 Ti                             88
OR094      06/08/2024 Sherie Minnock            Pending     NVIDIA GeForce GTX 1080                                99
           06/08/2024 Sherie Minnock            Pending     NVIDIA Quadro K8000 24GB                               19
OR197      05/08/2024 Anabal Jaxon              Pending     NVIDIA GeForce GTX 970                                 20
```



QUERY-LIST OF PENDING CASES (RESULT)

```
           03/10/2022 Evie Gaskill              Pending     NVIDIA GeForce GTX 1660 Ti                              7
OR356      22/09/2022 Bathsheba List            Pending     NVIDIA Quadro P6000 Max-Q 24GB                     67,321
           22/09/2022 Bathsheba List            Pending     NVIDIA Jetson AGX Orin                             78,129
OR459      19/09/2022 Eldredge Fletcher         Pending     NVIDIA GeForce GTX 1070 Ti                         96,726
           19/09/2022 Eldredge Fletcher         Pending     NVIDIA Tesla V100 SXM2                              9,404
OR316      07/09/2022 Bathsheba Littrik         Pending     NVIDIA Quadro 4000                                 84,935
           07/09/2022 Bathsheba Littrik         Pending     NVIDIA Titan Xp                                    71,795
OR225      20/08/2022 Darlleen Haberjam         Pending     NVIDIA GeForce GTX 760                                  2
OR259      09/08/2022 Wrab Batt                 Pending     NVIDIA DGX Station A100                            15,165
           09/08/2022 Wrab Batt                 Pending     NVIDIA GeForce GTX 1660                            64,820
           09/08/2022 Wrab Batt                 Pending     NVIDIA Quadro FX 1800                              55,293
OR021      07/08/2022 Rutter Prandy             Pending     NVIDIA GeForce RTX 4080 Ti                             37
           07/08/2022 Rutter Prandy             Pending     NVIDIA Quadro K4200                                    62
           07/08/2022 Rutter Prandy             Pending     NVIDIA Quadro P5000 Max-Q 16GB                         52
           07/08/2022 Rutter Prandy             Pending     NVIDIA Quadro P4000                                    60
OR362      27/07/2022 Garrott Durber            Pending     NVIDIA GeForce GTX 1660                            79,095
           27/07/2022 Garrott Durber            Pending     NVIDIA Quadro RTX 8000 SLI                         11,974
OR059      26/07/2022 Elna Swait                Pending     NVIDIA Quadro K5000 4GB                                66
           26/07/2022 Elna Swait                Pending     NVIDIA GeForce RTX 4090 Super                          22
OR149      22/07/2022 Xena Wayt                 Pending     NVIDIA GeForce GT 710                                   9
OR230      15/07/2022 Giuditta Gibson           Pending     NVIDIA Quadro P4000 Max-Q                              29
OR336      11/07/2022 Pearce Bywaters           Pending     NVIDIA GeForce GTX 760                              7,949
           11/07/2022 Pearce Bywaters           Pending     NVIDIA GeForce GTX 760 Ti 4GB                      53,231
OR043      01/07/2022 Geneva Dietsche           Pending     NVIDIA GeForce GT 710                                  18
           01/07/2022 Geneva Dietsche           Pending     NVIDIA GeForce GTX 980 4GB                             35
**********                                      -----------                                            ---------------
TotalUnit                                                                                                   6,223,408
```



DATABASE

INDIVIDUAL QUERY

PURPOSE: QUERY CUSTOMER TOTAL SPENDING

2024

Identifying Loyal Customers

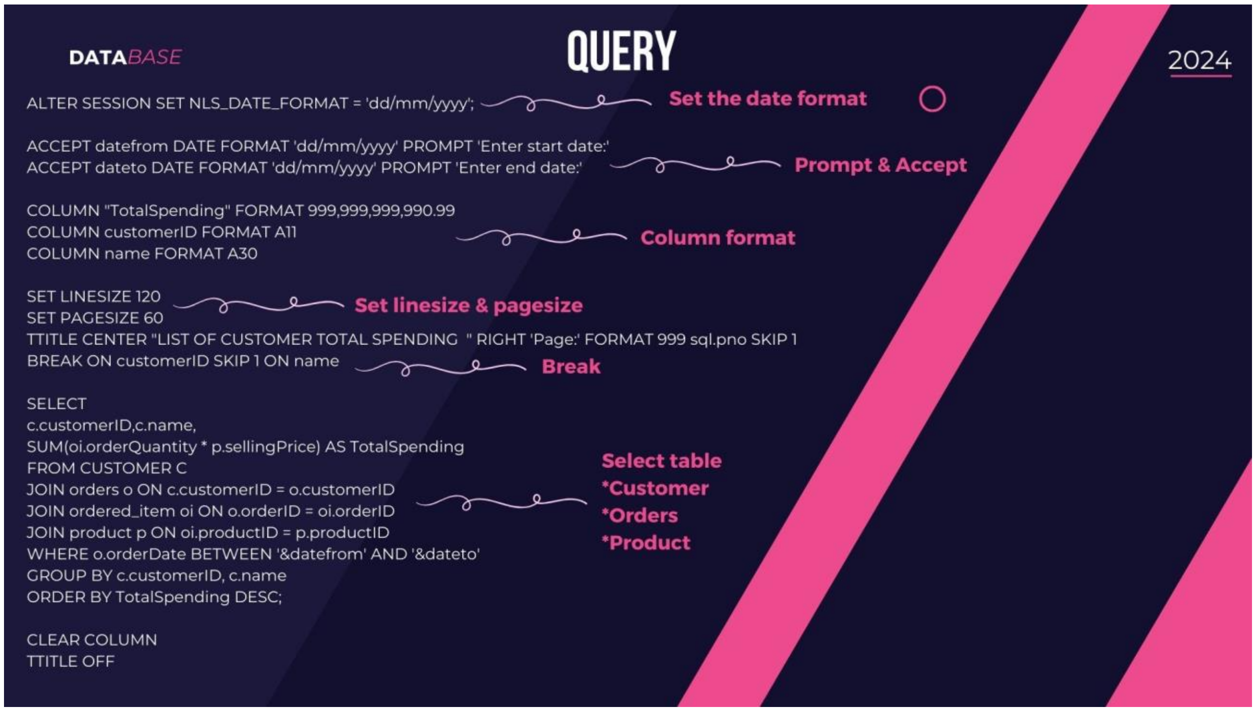
By tracking total spending over time, businesses can easily identify customers who consistently make purchases, indicating loyalty and engagement.

Calculating Customer Lifetime Value

Customer total spending is a key component in calculating customer lifetime value (CLTV), a metric that measures the long-term profitability of a customer relationship. By understanding CLTV, businesses can prioritize customer acquisition and retention efforts.

Personalized Marketing

By analyzing customer total spending, businesses can create personalized marketing campaigns that resonate with individual customers. This can lead to increased customer engagement and higher conversion rates.

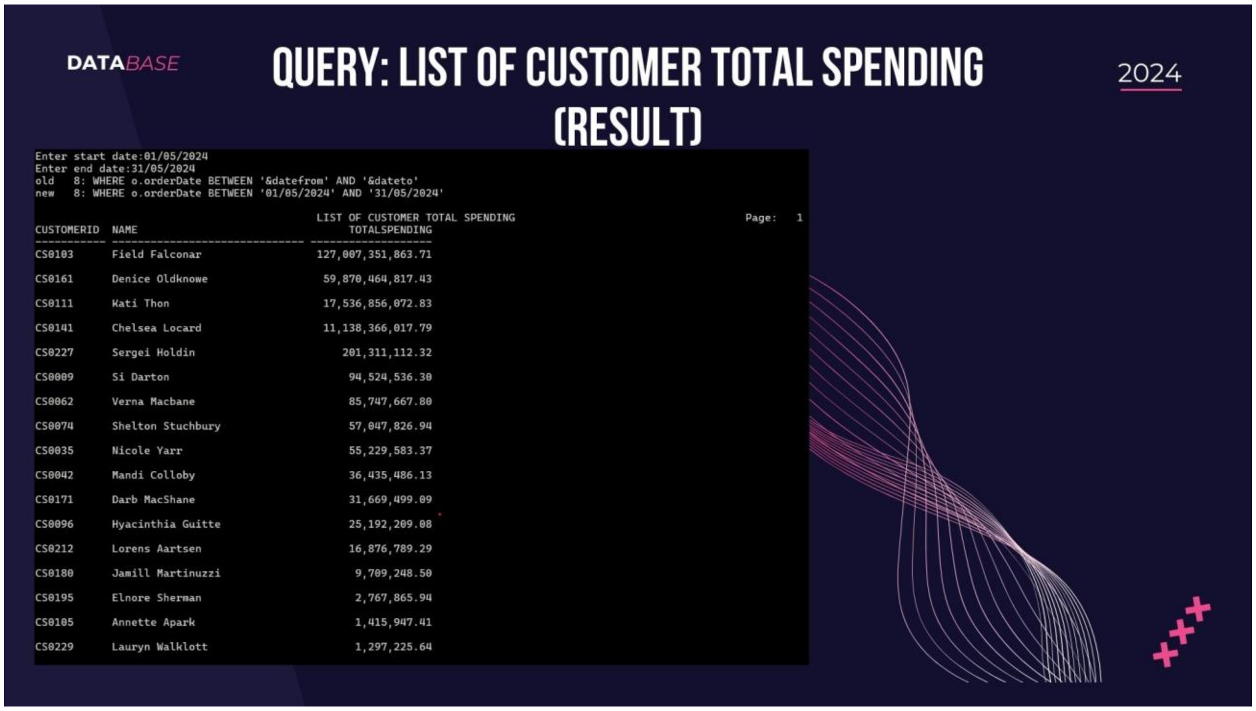


DATABASE

QUERY

2024

```
ALTER SESSION SET NLS_DATE_FORMAT = 'dd/mm/yyyy';

ACCEPT datefrom DATE FORMAT 'dd/mm/yyyy' PROMPT 'Enter start date:'
ACCEPT dateto DATE FORMAT 'dd/mm/yyyy' PROMPT 'Enter end date:'

COLUMN "TotalSpending" FORMAT 999,999,999,990.99
COLUMN customerID FORMAT A11
COLUMN name FORMAT A30

SET LINESIZE 120
SET PAGESIZE 60
TTITLE CENTER "LIST OF CUSTOMER TOTAL SPENDING  " RIGHT 'Page:' FORMAT 999 sql.pno SKIP 1
BREAK ON customerID SKIP 1 ON name

SELECT
c.customerID,c.name,
SUM(oi.orderQuantity * p.sellingPrice) AS TotalSpending
FROM CUSTOMER C
JOIN orders o ON c.customerID = o.customerID
JOIN ordered_item oi ON o.orderID = oi.orderID
JOIN product p ON oi.productID = p.productID
WHERE o.orderDate BETWEEN '&datefrom' AND '&dateto'
GROUP BY c.customerID, c.name
ORDER BY TotalSpending DESC;

CLEAR COLUMN
TTITLE OFF
```

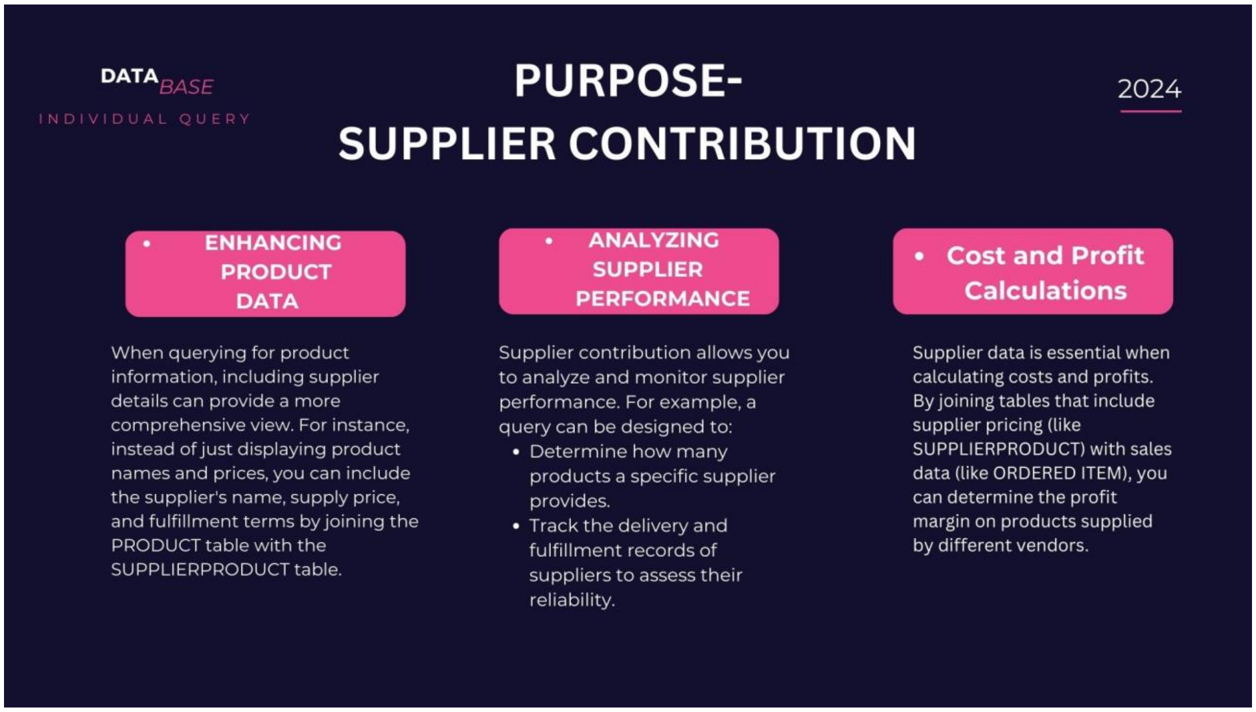


DATABASE

QUERY: LIST OF CUSTOMER TOTAL SPENDING (RESULT)

2024

```
Enter start date:01/05/2024
Enter end date:31/05/2024
old   8: WHERE o.orderDate BETWEEN '&datefrom' AND '&dateto'
new   8: WHERE o.orderDate BETWEEN '01/05/2024' AND '31/05/2024'

                                        LIST OF CUSTOMER TOTAL SPENDING                          Page:    1
CUSTOMERID  NAME                                TOTALSPENDING
----------- ------------------------------ -------------------
CS0103      Field Falconar                  127,007,351,863.71

CS0161      Denice Oldknowe                  59,870,464,817.43

CS0111      Kati Thon                        17,536,856,072.83

CS0141      Chelsea Locard                   11,138,366,017.79

CS0227      Sergei Holdin                       201,311,112.32

CS0009      Si Darton                            94,524,536.30

CS0062      Verna Macbane                        85,747,667.80

CS0074      Shelton Stuchbury                    57,047,826.94

CS0035      Nicole Yarr                          55,229,583.37

CS0042      Mandi Colloby                        36,435,486.13

CS0171      Darb MacShane                        31,669,499.09

CS0096      Hyacinthia Guitte                    25,192,209.08

CS0212      Lorens Aartsen                       16,876,789.29

CS0180      Jamill Martinuzzi                     9,709,248.50

CS0195      Elnore Sherman                        2,767,865.94

CS0105      Annette Apark                         1,415,947.41

CS0229      Lauryn Walklott                       1,297,225.64
```



DATABASE

INDIVIDUAL QUERY

PURPOSE- SUPPLIER CONTRIBUTION

2024

- **ENHANCING PRODUCT DATA**

When querying for product information, including supplier details can provide a more comprehensive view. For instance, instead of just displaying product names and prices, you can include the supplier's name, supply price, and fulfillment terms by joining the PRODUCT table with the SUPPLIERPRODUCT table.

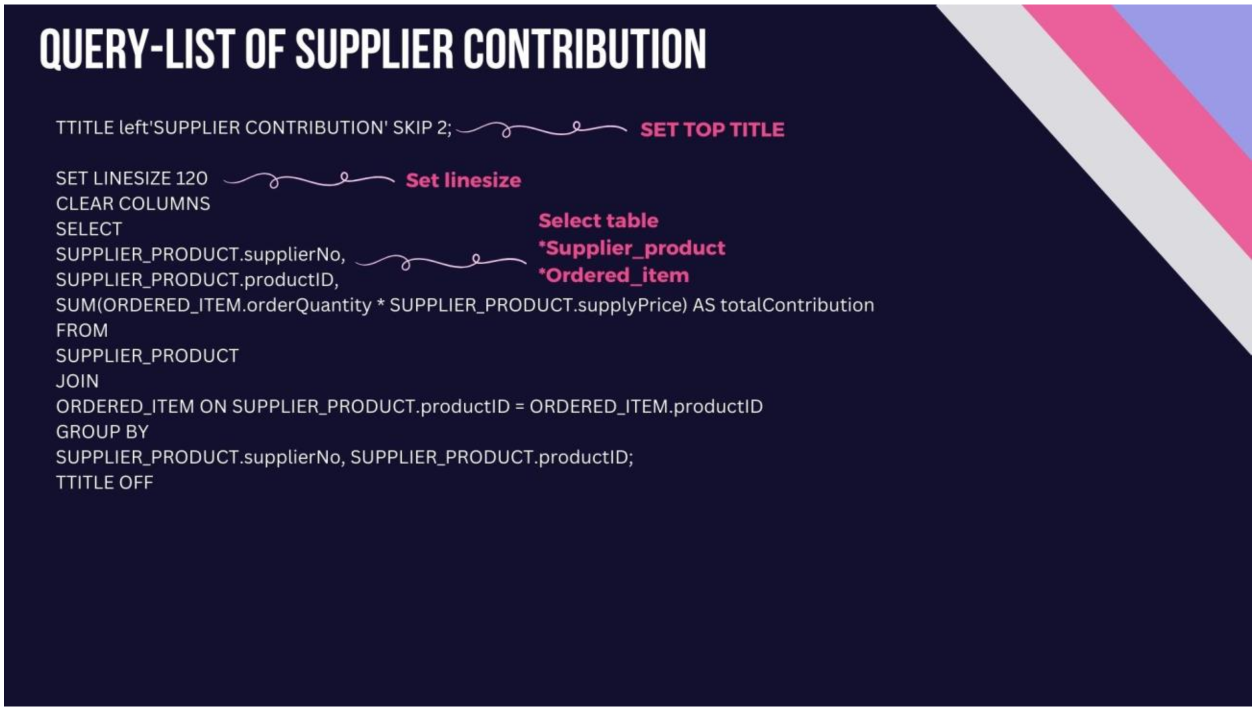
- **ANALYZING SUPPLIER PERFORMANCE**

Supplier contribution allows you to analyze and monitor supplier performance. For example, a query can be designed to:

- Determine how many products a specific supplier provides.
- Track the delivery and fulfillment records of suppliers to assess their reliability.

- **Cost and Profit Calculations**

Supplier data is essential when calculating costs and profits. By joining tables that include supplier pricing (like SUPPLIERPRODUCT) with sales data (like ORDERED ITEM), you can determine the profit margin on products supplied by different vendors.



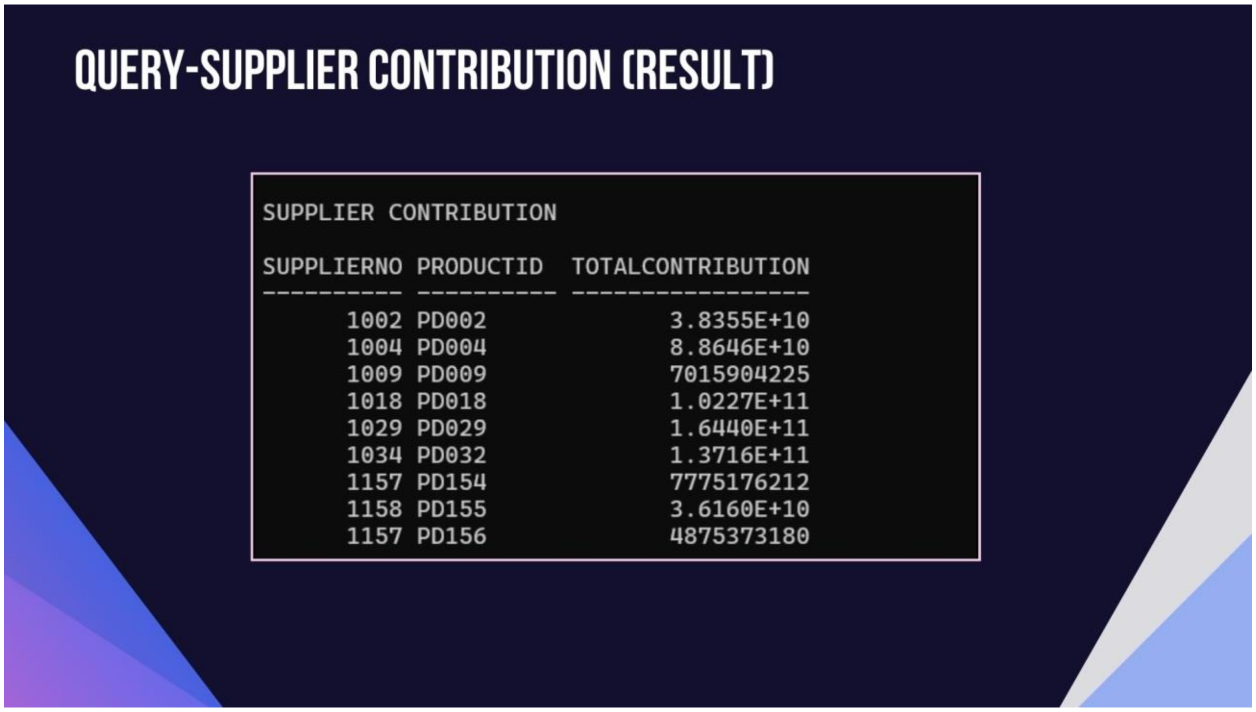
QUERY-LIST OF SUPPLIER CONTRIBUTION

```
TTITLE left'SUPPLIER CONTRIBUTION' SKIP 2;
SET LINESIZE 120
CLEAR COLUMNS
SELECT
SUPPLIER_PRODUCT.supplierNo,
SUPPLIER_PRODUCT.productID,
SUM(ORDERED_ITEM.orderQuantity * SUPPLIER_PRODUCT.supplyPrice) AS totalContribution
FROM
SUPPLIER_PRODUCT
JOIN
ORDERED_ITEM ON SUPPLIER_PRODUCT.productID = ORDERED_ITEM.productID
GROUP BY
SUPPLIER_PRODUCT.supplierNo, SUPPLIER_PRODUCT.productID;
TTITLE OFF
```

SET TOP TITLE

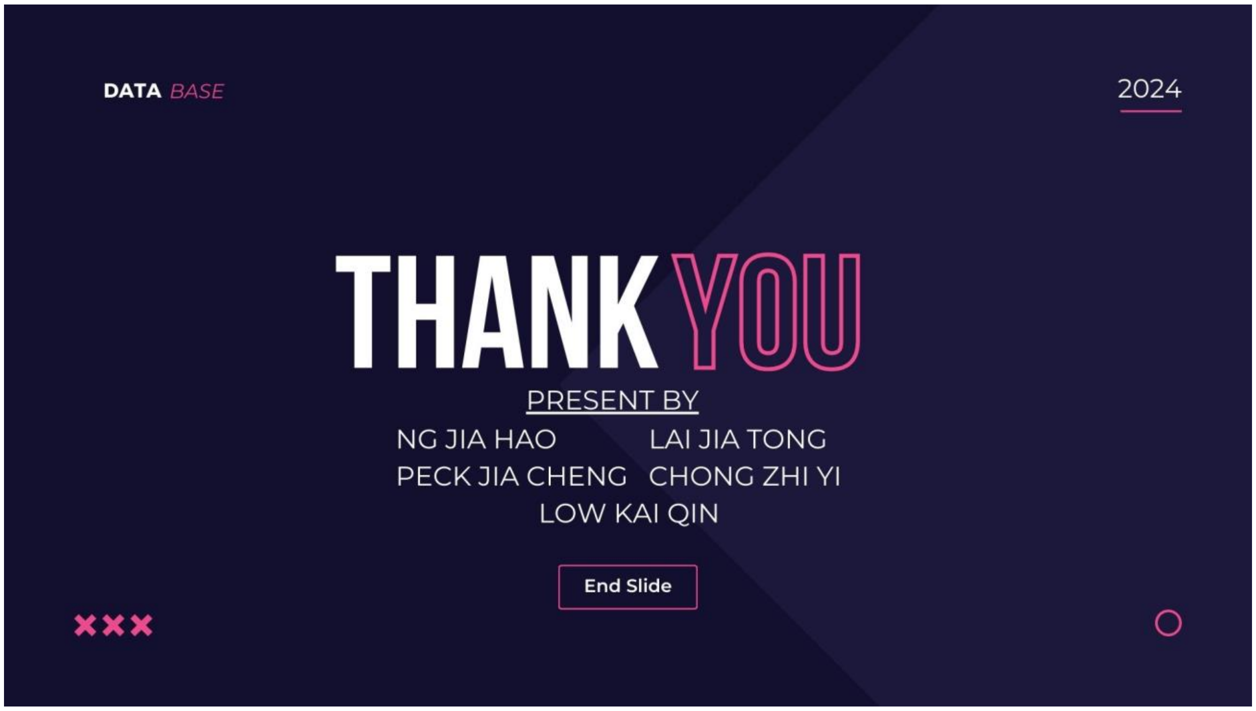
Set linesize

Select table
*Supplier_product
*Ordered_item



QUERY-SUPPLIER CONTRIBUTION (RESULT)

```
SUPPLIER CONTRIBUTION

SUPPLIERNO PRODUCTID  TOTALCONTRIBUTION
---------- ---------- -----------------
      1002 PD002             3.8355E+10
      1004 PD004             8.8646E+10
      1009 PD009             7015904225
      1018 PD018             1.0227E+11
      1029 PD029             1.6440E+11
      1034 PD032             1.3716E+11
      1157 PD154             7775176212
      1158 PD155             3.6160E+10
      1157 PD156             4875373180
```



DATA BASE

2024

THANK YOU

PRESENT BY

NG JIA HAO LAI JIA TONG

PECK JIA CHENG CHONG ZHI YI

LOW KAI QIN

End Slide