



ADVENTURE WORKS – MYSQL PROJECT

Team Members Names:

Gudala Satvika

K.Kishan

K.Pradeepthi

P Sai Nitya

Contents of the Presentation:

1.Overview

2.Tables

3.My Sql queries

4.My Sql script images

5.Thank you

OVERVIEW

This project focused on analyzing sales data from the Adventure Works sample database. You likely used SQL queries to extract and analyze this data, uncovering valuable insights to inform business decisions.

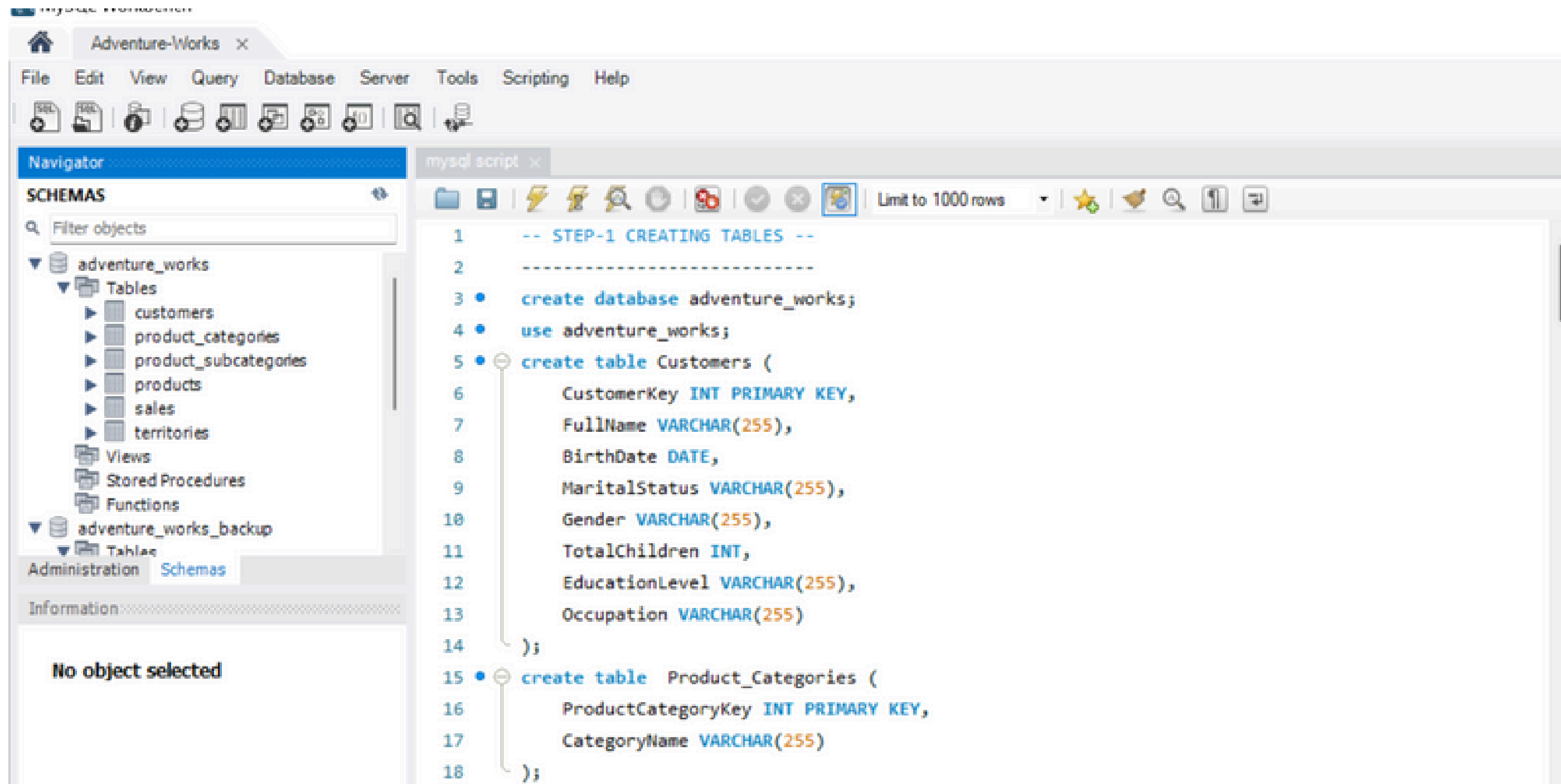
- Understand customer buying behavior and product performance

- Identify profitable products and categories

- Analyze sales trends and regional variations

- Gain insights to optimize marketing strategies and resource allocation

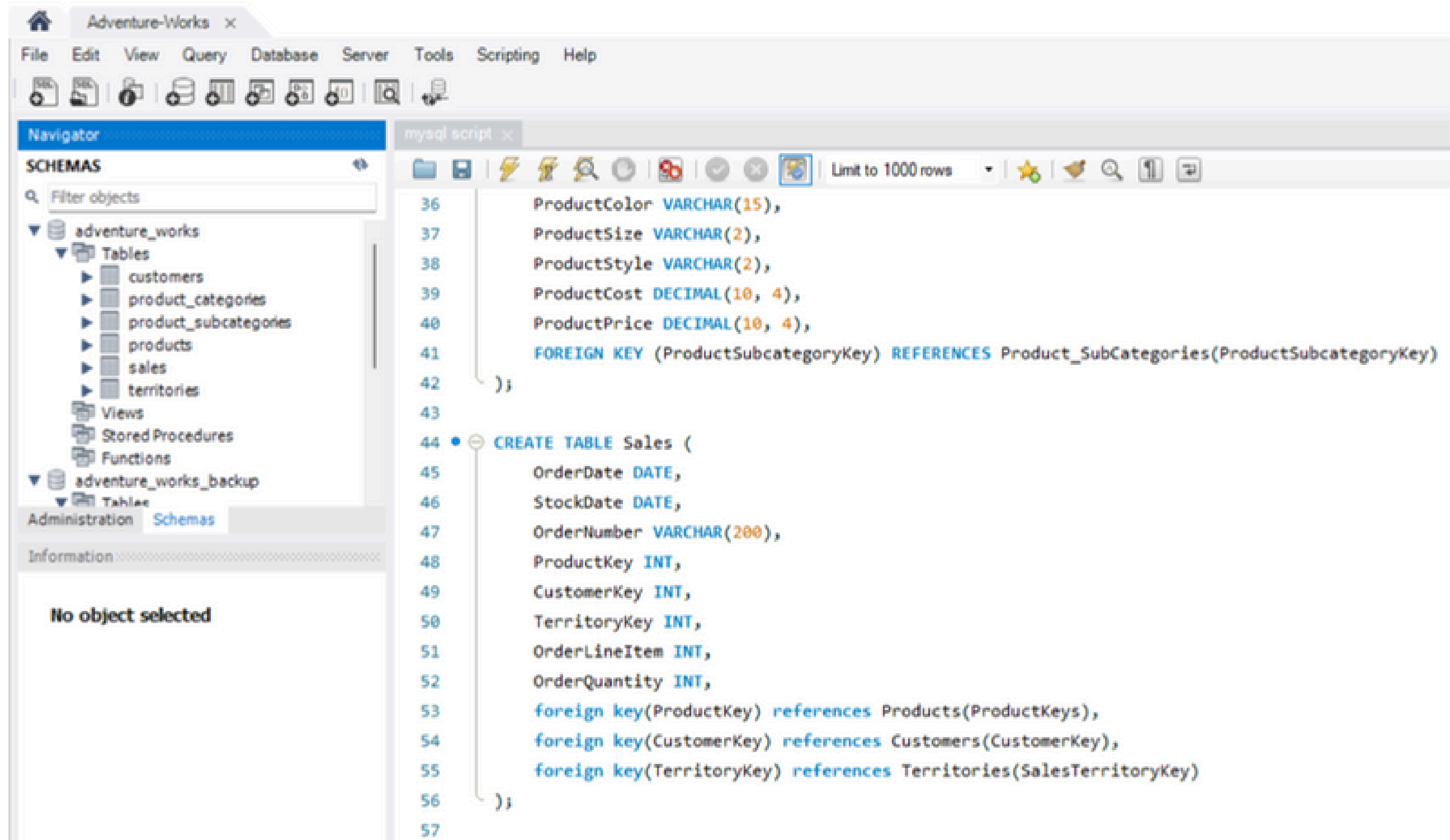
TABLES



SLIDE#5

The screenshot displays the SQL Server Enterprise Manager interface with the Adventure-Works database selected. The left pane shows the database structure, including tables like customers, product_categories, product_subcategories, products, sales, and territories. The right pane shows a MySQL script editor with the following SQL code:

```
19 • create table Product_SubCategories (  
20     ProductSubcategoryKey INT PRIMARY KEY,  
21     SubcategoryName VARCHAR(255),  
22     ProductCategoryKey INT,  
23     FOREIGN KEY (ProductcategoryKey) REFERENCES Product_Categories(ProductcategoryKey)  
24 );  
25 • create table Territories (  
26     SalesTerritoryKey INT PRIMARY KEY,  
27     Region VARCHAR(255),  
28     Country VARCHAR(255),  
29     Continent VARCHAR(255)  
30 );  
31 • CREATE TABLE Products (  
32     ProductKeys INT PRIMARY KEY,  
33     ProductSubcategoryKey INT,  
34     ProductName VARCHAR(50),  
35     ModelName VARCHAR(30),  
36     ProductColor VARCHAR(15),  
37     ProductSize VARCHAR(2),  
38     ProductStyle VARCHAR(2),  
39     ProductCost DECIMAL(10, 4),  
40     ProductPrice DECIMAL(10, 4),
```



The screenshot displays the SQL Server Enterprise Manager interface. The left pane shows the 'Adventure-Works' database with its schema structure. The right pane shows a SQL script for creating the 'Sales' table, which includes foreign key references to other tables in the database.

Navigator

SCHEMAS

Filter objects

- adventure_works
 - Tables
 - customers
 - product_categories
 - product_subcategories
 - products
 - sales
 - territories
 - Views
 - Stored Procedures
 - Functions
- adventure_works_backup
 - Tables

Administration Schemas

Information

No object selected

mysql script x

Limit to 1000 rows

```
36 ProductColor VARCHAR(15),
37 ProductSize VARCHAR(2),
38 ProductStyle VARCHAR(2),
39 ProductCost DECIMAL(10, 4),
40 ProductPrice DECIMAL(10, 4),
41 FOREIGN KEY (ProductSubcategoryKey) REFERENCES Product_SubCategories(ProductSubcategoryKey)
42 );
43
44 CREATE TABLE Sales (
45 OrderDate DATE,
46 StockDate DATE,
47 OrderNumber VARCHAR(200),
48 ProductKey INT,
49 CustomerKey INT,
50 TerritoryKey INT,
51 OrderLineItem INT,
52 OrderQuantity INT,
53 foreign key(ProductKey) references Products(ProductKeys),
54 foreign key(CustomerKey) references Customers(CustomerKey),
55 foreign key(TerritoryKey) references Territories(SalesTerritoryKey)
56 );
57
```

The screenshot displays the SQL Server Enterprise Manager interface. The left pane shows the 'Adventure-Works' database schema, with the 'Tables' folder expanded under 'Schemas'. The tables listed are customers, product_categories, product_subcategories, products, sales, and territories. The right pane shows a 'mysql script' window with the following SQL code:

```
56 );
57
58
59
60 -- STEP-2 CREATING BACKUP DATABASE AND IMPORTING THE TABLES--
61 -----
62 • create database adventure_works_backup;
63 • use adventure_works_backup;
64
65 • set sql_safe_updates = 0;
66 • update aw_customers set BirthDate= str_to_date(BirthDate, '%Y-%m-%d');
67 • ALTER TABLE aw_customers modify birthdate date;
68 • update aw_sales set OrderDate= str_to_date(OrderDate, '%Y-%m-%d');
69 • ALTER TABLE aw_sales modify Orderdate date;
70 • update aw_sales set StockDate= str_to_date(StockDate, '%Y-%m-%d');
71 • ALTER TABLE aw_sales modify Stockdate date;
```

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Schemas' tree shows the 'adventure_works' database expanded, listing tables like customers, product_categories, product_subcategories, products, sales, and territories. Below it, 'adventure_works_backup' is also expanded. The main window, titled 'mysql script', contains the following SQL script:

```
69 • ALTER TABLE aw_sales modify Orderdate date;
70 • update aw_sales set StockDate= str_to_date(StockDate, '%Y-%m-%d');
71 • ALTER TABLE aw_sales modify Stockdate date;
72
73 -- INSERTING TABLES FROM BACKUP TO MAIN DATABASE --
74 -----
75 • INSERT INTO adventure_works.customers select * from adventure_works_backup.aw_customers;
76 • INSERT INTO adventure_works.product_categories select * from adventure_works_backup.aw_product_cat
77 • INSERT INTO adventure_works.product_subcategories select * from adventure_works_backup.aw_product_
78 • INSERT INTO adventure_works.products select * from adventure_works_backup.aw_products;
79 • INSERT INTO adventure_works.territories select * from adventure_works_backup.aw_territories;
80 • INSERT INTO adventure_works.sales select * from adventure_works_backup.aw_sales;
81
82
```


SLIDE#9

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Schemas' tree for the 'adventure_works_backup' database, listing tables like 'aw_customers', 'aw_product_categories', and 'aw_products'. The right pane shows a 'mysql script' window with a query titled '-- STEP-3 CREATING QUERIES --'. The query is a SELECT statement that calculates the total revenue by product category. The 'Result Grid' at the bottom displays the results of the query.

```
-- STEP-3 CREATING QUERIES --
-----
-- 1. Total Revenue by Product Category.
SELECT pc.CategoryName AS Product_Category,
       SUM(s.OrderQuantity * p.ProductPrice) AS Total_Revenue
FROM Sales s
JOIN Products p ON s.ProductKey = p.ProductKeys
JOIN Product_SubCategories ps ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
JOIN Product_Categories pc ON ps.ProductCategoryKey = pc.ProductCategoryKey
GROUP BY pc.CategoryName;
```

Product_Category	Total_Revenue
Bikes	11208594.9379
Clothing	133367.2854
Accessories	325592.3916

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Schemas' tree for the 'adventure_works_backup' database. The right pane shows a 'mysql script' window with a query titled '-- 2. Total Revenue by Product Category and Region.'. The query is a SELECT statement that calculates the total revenue by product category and region. The 'Result Grid' at the bottom displays the results of the query.

```
-- 2. Total Revenue by Product Category and Region.
SELECT pc.CategoryName AS Product_Category,
       t.Region,
       SUM(s.OrderQuantity * p.ProductPrice) AS Total_Revenue
FROM Sales s
JOIN Products p ON s.ProductKey = p.ProductKeys
JOIN Territories t ON s.TerritoryKey = t.SalesTerritoryKey
JOIN Product_SubCategories ps ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
JOIN Product_Categories pc ON ps.ProductCategoryKey = pc.ProductCategoryKey
GROUP BY pc.CategoryName, t.Region;
```

Product_Category	Region	Total_Revenue
Accessories	Germany	30563.2448
Accessories	France	29291.9154
Accessories	United Kingdom	40567.6548
Accessories	Australia	70458.7362
Accessories	Canada	50695.6014
Accessories	Southwest	59376.0080
Accessories	Northwest	44134.1242
Accessories	Northeast	330000.0000

SLIDE#10

Adventure-Works x

File Edit View Query Database Server Tools Scripting Help

Navigator

mysql script x

Limit to 1000 rows

103 GROUP BY pc.CategoryName, t.Region;

104

105 -- 3. Average Order Value by Product Category.

106 • SELECT pc.CategoryName AS Product_Category,

107 AVG(s.OrderQuantity * p.ProductPrice) AS Average_Order_Value

108 FROM Sales s

109 JOIN Products p ON s.ProductKey = p.ProductKeys

110 JOIN Product_SubCategories ps ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey

111 JOIN Product_Categories pc ON ps.ProductCategoryKey = pc.ProductCategoryKey

112 GROUP BY pc.CategoryName;

113

Result Grid

Product_Category	Average_Order_Value
Bikes	1885.69901378
Accessories	27.73831927
Clothing	43.24490447

Result 3 x

Read Only

Adventure-Works x

File Edit View Query Database Server Tools Scripting Help

Navigator

mysql script x

Limit to 1000 rows

113

114 -- 4. Daily Sales Trend.

115 • SELECT s.OrderDate, SUM(p.ProductPrice * s.OrderQuantity) AS Daily_Sales

116 FROM Sales s

117 JOIN Products p ON s.ProductKey = p.ProductKeys

118 GROUP BY s.OrderDate

119 ORDER BY s.OrderDate;

120

121 -- 5. Profit by Products.

122 • SELECT p.ProductName,

123 SUM((p.ProductPrice - p.ProductCost) * s.OrderQuantity) AS Profit

Result Grid

OrderDate	Daily_Sales
2015-01-01	4074.0882
2015-01-02	10734.8100
2015-01-03	17509.7900
2015-01-04	6978.2600
2015-01-05	699.0982
2015-01-07	7156.5400
2015-01-08	11255.6282
2015-01-09	2578.3200

Result 4 x

Read Only

SLIDE#11

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Schemas' tree for the 'adventure_works_backup' database, listing tables like 'aw_customers', 'aw_product_categories', 'aw_product_subcategories', 'aw_products', 'aw_sales', 'aw_territories', and 'customers'. The right pane shows a query window with the following SQL code:

```
118 GROUP BY s.OrderDate
119 ORDER BY s.OrderDate;
120
121 -- 5. Profit by Products.
122 • SELECT p.ProductName,
123        SUM((p.ProductPrice - p.ProductCost) * s.OrderQuantity) AS Profit
124 FROM Sales s
125 JOIN Products p ON s.ProductKey = p.ProductKeys
126 GROUP BY p.ProductName;
127
128 -- 6. Profit by Product Sub-Categories.
```

The 'Result Grid' at the bottom displays the results of the query:

ProductName	Profit
Road-650 Black, 58	10580.2203
Mountain-100 Black, 44	26584.1208
Road-150 Red, 48	146325.4832
Road-150 Red, 56	123813.8704
Mountain-100 Silver, 42	25293.2052
Road-150 Red, 52	130848.7494
Mountain-100 Silver, 38	25293.2052

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Schemas' tree for the 'adventure_works_backup' database, listing tables like 'aw_customers', 'aw_product_categories', 'aw_product_subcategories', 'aw_products', 'aw_sales', 'aw_territories', and 'customers'. The right pane shows a query window with the following SQL code:

```
126 GROUP BY p.ProductName;
127
128 -- 6. Profit by Product Sub-Categories.
129 • SELECT ps.SubcategoryName,
130        SUM((p.ProductPrice - p.ProductCost) * s.OrderQuantity) AS Profit
131 FROM Sales s
132 JOIN Products p ON s.ProductKey = p.ProductKeys
133 JOIN Product_SubCategories ps ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
134 GROUP BY ps.SubcategoryName;
135
136 -- 7. Total Customers.
```

The 'Result Grid' at the bottom displays the results of the query:

SubcategoryName	Profit
Road Bikes	1920763.8419
Mountain Bikes	2413770.0808
Bottles and Cages	23389.1812
Helmets	45435.2600
Jerseys	16882.6174
Tires and Tubes	73700.0756
Fenders	31660.6095

SLIDE#12

Adventure-Works

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

adventure_works_backup

Tables

- aw_customers
- aw_product_categories
- aw_product_subcategories
- aw_products
- aw_sales
- aw_territories
- customers

Views

Stored Procedures

Functions

Administration Schemas

Information

Result Grid

Filter Rows:

Exports

Wrap Cell Content:

No object selected

Result 7

Read Only

```
133 JOIN Product_SubCategories ps ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
134 GROUP BY ps.SubcategoryName;
135
136 -- 7. Total Customers.
137 • SELECT COUNT(*) AS Total_Customers
138 FROM Customers;
139
140 -- 8. Product Sub-category Sales Contribution
141 • SELECT ps.SubcategoryName,
142 SUM(s.OrderQuantity * p.ProductPrice) AS SalesContribution
143 FROM Sales s
```

Total_Customers
8348

Adventure-Works

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHMAS

Filter objects

adventure_works_backup

Tables

- aw_customers
- aw_product_categories
- aw_product_subcategories
- aw_products
- aw_sales
- aw_territories
- customers

Views

Stored Procedures

Functions

Administration Schemas

Information

Result Grid

Filter Rows:

Exports

Wrap Cell Content:

No object selected

Result 8

Read Only

```
138 FROM Customers;
139
140 -- 8. Product Sub-category Sales Contribution
141 • SELECT ps.SubcategoryName,
142 SUM(s.OrderQuantity * p.ProductPrice) AS SalesContribution
143 FROM Sales s
144 JOIN Products p ON s.ProductKey = p.ProductKeys
145 JOIN Product_SubCategories ps ON p.ProductSubcategoryKey = ps.ProductSubcategoryKey
146 GROUP BY ps.SubcategoryName;
147
148 -- 9. Products with Highest Profit.
```

SubcategoryName	SalesContribution
Road Bikes	4934501.3869
Mountain Bikes	5271451.1610
Bottles and Cages	37363.2400
Helmets	71384.9416
Jerseys	59034.6925
Tires and Tubes	117732.2800
Fenders	50575.9800

SLIDE#13

Adventure-Works x

File Edit View Query Database Server Tools Scripting Help

Navigator

mysql script x

Limit to 1000 rows

```
147
148 -- 9. Products with Highest Profit.
149 • SELECT p.ProductName,
150       SUM((p.ProductPrice - p.ProductCost) * s.OrderQuantity) AS Profit
151 FROM Sales s
152 JOIN Products p ON s.ProductKey = p.ProductKeys
153 GROUP BY p.ProductName
154 ORDER BY Profit DESC
155 LIMIT 10;
156
157 -- 10. Territory-wise Sales.
```

SCHEMAS

Filter objects

adventure_works_backup

- Tables
 - aw_customers
 - aw_product_categories
 - aw_product_subcategories
 - aw_products
 - aw_sales
 - aw_territories
 - customers
- Views
- Stored Procedures
- Functions

Administration Schemas

Information

No object selected

Result Grid

ProductName	Profit
Mountain-200 Black, 46	367882.3980
Mountain-200 Black, 42	341470.3284
Mountain-200 Silver, 38	341375.8046
Mountain-200 Black, 38	329207.5818
Mountain-200 Silver, 46	328979.4765
Mountain-200 Silver, 42	324211.6580
Road-150 Red, 48	146325.4832
Road-150 Red, 42	133755.3323

Result 9 x

Read Only

Adventure-Works x

File Edit View Query Database Server Tools Scripting Help

Navigator

mysql script x

Limit to 1000 rows

```
155 LIMIT 10;
156
157 -- 10. Territory-wise Sales.
158 • SELECT t.Region,
159       SUM(s.OrderQuantity * p.ProductPrice) AS Total_Sales
160 FROM Sales s
161 JOIN Products p ON s.ProductKey = p.ProductKeys
162 JOIN Territories t ON s.TerritoryKey = t.SalesTerritoryKey
163 GROUP BY t.Region;
164
165
```

SCHEMAS

Filter objects

adventure_works_backup

- Tables
 - aw_customers
 - aw_product_categories
 - aw_product_subcategories
 - aw_products
 - aw_sales
 - aw_territories
 - customers
- Views
- Stored Procedures
- Functions

Administration Schemas

Information

No object selected

Result Grid

Region	Total_Sales
Northwest	1493154.8561
Northeast	2032.4788
Central	763.9004
Southwest	2119084.6370
Southeast	1583.1345
Canada	462729.4006
France	1403786.6887
Germany	137001.6370

Result 10 x

Read Only

INSIGHTS

You can gain valuable insights into your product performance, customer behavior, and overall sales effectiveness. This information can be used to make data-driven decisions to improve profitability, marketing strategies, and resource allocation within your company.

Expected Outcomes:

- Data-driven insights on product performance, customer preferences, and sales trends

 - Identification of high-profit products and categories

 - Understanding of regional sales variations

 - Information to optimize marketing campaigns and resource allocation



THANK YOU

<https://github.com/GudalaSatvika>



UNITED NETWORK
OF PROFESSIONALS

The Future Of Effective Learning

UNP is a decentralized education platform enabling
quality education for all

www.unp.education | reachout@unp.education
reachoutunp@gmail.com | +1 929-288-1787

CONNECT WITH US

