Lab 1

1. **Write a SELECT statement that lists the customers along with their ID numbers. Include the last names, first names, and company names.**
   1. SELECT CustomerID, LastName, FirstName, CompanyName FROM SalesLT.Customer;
2. **Write a SELECT statement that lists the name, product number, and color of each product**
   1. SELECT Name, ProductNumber, Color FROM SalesLT.Product;
3. **Write a SELECT statement that lists the customer ID numbers and sales order ID numbers from the SalesLT.SalesOrderHeader table**
   1. SELECT CustomerID, SalesOrderID FROM SalesLT.SalesOrderHeader;
4. **Write a query that displays all the employees listed in the HumanResources.Employee table who have the job title Research and Development Engineer. Display the business entity ID number, the login ID, and the title for each one**
   1. SELECT BusinessEntityID, JobTitle, LoginID FROM HumanResources.Employee WHERE JobTitle = 'Research and Development Engineer';
5. **Write a query that displays all the names in Person.Person with the middle name J. Display the first, last, and middle names along with the ID numbers**
   1. SELECT FirstName, MiddleName, LastName, BusinessEntityID FROM Person.Person WHERE MiddleName = 'J';
6. **Write a query displaying all the columns of the Production.ProductCostHistory table from the rows that were modified on June 17, 2003.**
   1. SELECT [ProductID] ,[StartDate] ,[EndDate] ,[StandardCost] ,[ModifiedDate] FROM [AdventureWorks2008].[Production].[ProductCostHistory] WHERE ModifiedDate = '2003-06-17';
7. **Write a query that displays all the rows from the Person.Person table where the rows were modified after December 29, 2000. Display the business entity ID number, the name columns, and the modified date.**
   1. SELECT BusinessEntityID, FirstName, MiddleName, LastName, ModifiedDate

FROM Person.Person WHERE ModifiedDate > '2000-12-29';

1. **Rewrite the last query so that the rows that were not modified on December 29, 2000, are displayed**
   1. SELECT BusinessEntityID, FirstName, MiddleName, LastName, ModifiedDate

FROM Person.Person WHERE ModifiedDate <> '2000-12-29';

1. **Write a query that displays all the rows from the Person.Person table where the rows were modified during December 2000.**
   1. SELECT BusinessEntityID, FirstName, MiddleName, LastName, ModifiedDate

FROM Person.Person WHERE ModifiedDate BETWEEN '2000-12-01' AND '2000-12-31';

1. **Write a query that displays the product ID and name for each product from the Production.Product table with the name starting with Chain.**
   1. SELECT ProductID, Name FROM Production.Product WHERE Name LIKE 'Chain%'
2. **Write a query that displays the products with helmet in the name**
   1. SELECT ProductID, Name FROM Production.Product WHERE Name LIKE '%helmet%';
3. **Change the last query so that the products without helmet in the name are displayed**
   1. SELECT ProductID, Name FROM Production.Product WHERE Name NOT LIKE '%helmet%';
4. **Write a query that displays the business entity ID number, first name, middle name, and last name from the Person.Person table for only those rows that have E or B stored in the middle name column**
   1. SELECT BusinessEntityID, FirstName, MiddleName, LastName FROM Person.Person

WHERE MiddleName LIKE '[E,B]';

1. **Explain the difference between the following two queries:** 
   1. **SELECT FirstName FROM Person.Person WHERE LastName LIKE 'Ja%es';**
   2. **SELECT FirstName FROM Person.Person WHERE LastName LIKE 'Ja\_es**

The first query will return rows with any number of characters replacing the percent sign. The second query will allow only one character to replace the underscore character

1. **Write a query displaying the order ID, order date, and total due from the Sales.SalesOrderHeadertable. Retrieve only those rows where the order was placed during the month of September 2001 and the total due exceeded $1,000.**
   1. SELECT SalesOrderID, OrderDate, TotalDue FROM Sales.SalesOrderHeader WHERE OrderDate BETWEEN '2001-09-01' AND '2001-09-30' AND TotalDue > 1000;
2. **Write a query displaying the sales orders where the total due exceeds $1,000. Retrieve only those rows where the salesperson ID is 279 or the territory ID is 6**
   1. SELECT SalesOrderID, OrderDate, TotalDue, SalesPersonID, TerritoryID FROM Sales.SalesOrderHeader WHERE TotalDue > 1000 AND (SalesPersonID = 279 OR TerritoryID = 6);
3. **Write a query displaying the sales orders where the total due exceeds $1,000. Retrieve only those rows where the salesperson ID is 279 or the territory ID is 6 or 4.**
   1. SELECT SalesOrderID, OrderDate, TotalDue, SalesPersonID, TerritoryID FROM Sales.SalesOrderHeader WHERE TotalDue > 1000 AND (SalesPersonID = 279 OR TerritoryID IN (6,4));
4. **Write a query displaying the ProductID, Name, and Color columns from rows in the Production.Product table. Display only those rows where no color has been assigned**
   1. SELECT ProductID, Name, Color FROM Production.Product WHERE Color IS NULL;
5. **Write a query displaying the ProductID, Name, and Color columns from rows in the Production.Product table. Display only those rows in which the color is not blue**
   1. SELECT ProductID, Name, Color FROM Production.Product WHERE Color IS NULL OR Color <> 'Blue';
6. **Write a query displaying ProductID, Name, Style, Size, and Color from the Production.Producttable. Include only those rows where at least one of the Style, Size, or Color columns contains a value**
   1. SELECT ProductID, Name, Style, Size, Color FROM Production.Product WHERE Style IS NOT NULL OR Size IS NOT NULL OR Color IS NOT NULL;
7. **Write a query using the Production.ProductReview table. Find all the rows that have the word socks in the Comments column. Return the ProductID and Comments columns.**
   1. SELECT Comments,ProductID FROM Production.ProductReview WHERE CONTAINS(Comments,'socks');
8. **Write a query using the Production.Document table. Find all the rows that have the word reflector in any column that is indexed with Full-Text Search. Display the Title and FileName columns**
   1. SELECT Title,FileName FROM Production.Document WHERE CONTAINS (\*,'reflector');
9. **Change the previous query so that the rows containing seat are not returned in the results.**
   1. SELECT Title, FileName FROM Production.Document WHERE CONTAINS(\*,'reflector AND NOT seat')
10. **Write a query that returns the business entity ID and name columns from the Person.Persontable. Sort the results by LastName, FirstName, and MiddleName**
    1. SELECT BusinessEntityID, LastName, FirstName, MiddleName FROM Person.Person ORDER BY LastName, FirstName, MiddleName;
11. **Write a query that displays in the “AddressLine1 (City PostalCode)” format from the Person.Address table**
    1. SELECT AddressLine1 + ' (' + City + ' ' + PostalCode + ')' FROM Person.Address;
12. **Write a query using the Production.Product table displaying the product ID, color, and name columns. If the color column contains a NULL value, replace the color with No Color.**
    1. SELECT ProductID, ISNULL(Color,'No Color') AS Color, Name FROM Production.Product;
13. **Write a query using the Production.Product table displaying a description with the “ProductID: Name” format**
    1. SELECT CAST(ProductID AS VARCHAR) + ': ' + Name AS IDName FROM Production.Product;
14. **Write a query using the Sales.SpecialOffer table. Display the difference between the MinQty and MaxQty columns along with the SpecialOfferID and Description columns.**
    1. SELECT SpecialOfferID, Description, MaxQty - MinQty AS Diff FROM Sales.SpecialOffer;
15. **Write a query using the Sales.SpecialOffer table. Multiply the MinQty column by the DiscountPctcolumn. Include the SpecialOfferID and Description columns in the results**
    1. SELECT SpecialOfferID, Description, MinQty \* DiscountPct AS Discount FROM Sales.SpecialOffer;
16. **Write a query using the Sales.SpecialOffer table that multiplies the MaxQty column by the DiscountPCT column. If the MaxQty value is null, replace it with the value 10. Include the SpecialOfferID and Description columns in the results.**
    1. SELECT SpecialOfferID, Description, ISNULL(MaxQty,10) \* DiscountPct AS Discount FROM Sales.SpecialOffer;
17. **Write a query that displays the first 10 characters of the AddressLine1 column in the Person.Address table.**
    1. SELECT LEFT(AddressLine1,10) AS Address10 FROM Person.Address;
18. **Write a query that displays characters 10 to 15 of the AddressLine1 column in the Person.Addresstable.**
    1. SELECT SUBSTRING(AddressLine1,10,6) AS Address10to15 FROM Person.Address;
19. **Write a query displaying the first name and last name from the Person.Person table all in uppercase**
    1. SELECT UPPER(FirstName) AS FirstName, UPPER(LastName) AS LastName FROM Person.Person;
20. **The product number in the Production.Product contains a hyphen (-). Write a query that uses the SUBSTRING function and the CHARINDEX function to display the characters in the product number following the hyphen. Note: there is also a second hyphen in many of the rows; ignore the second hyphen for this question.**
    1. **--Step 1**

SELECT ProductNumber, CHARINDEX('-',ProductNumber) FROM Production.Product;

* 1. **--Step 2**

SELECT ProductNumber, SUBSTRING(ProductNumber,CHARINDEX('-',ProductNumber)+1,25) AS ProdNumber FROM Production.Product**;**

1. **Write a query that calculates the number of days between the date an order was placed and the date that it was shipped using the Sales.SalesOrderHeader table. Include the SalesOrderID, OrderDate, and ShipDate columns.**
   1. SELECT SalesOrderID, OrderDate, ShipDate, DATEDIFF(d,OrderDate,ShipDate) AS NumberOfDays FROM Sales.SalesOrderHeader;
2. **Write a query that displays only the date, not the time, for the order date and ship date in the Sales.SalesOrderHeader table**
   1. SELECT CONVERT(VARCHAR,OrderDate,1) AS OrderDate, CONVERT(VARCHAR, ShipDate,1) AS ShipDate FROM Sales.SalesOrderHeader;
3. **Write a query that adds six months to each order date in the Sales.SalesOrderHeader table. Include the SalesOrderID and OrderDate columns.**
   1. SELECT SalesOrderID, OrderDate, DATEADD(m,6,OrderDate) Plus6Months FROM Sales.SalesOrderHeader;
4. **Write a query that displays the year of each order date and the numeric month of each order date in separate columns in the results. Include the SalesOrderID and OrderDate columns.**
   1. SELECT SalesOrderID, OrderDate, DATEPART(yyyy,OrderDate) AS OrderYear,

DATEPART(m,OrderDate) AS OrderMonth FROM Sales.SalesOrderHeader**;**

1. **Change the above query to display the month name instead.**
   1. SELECT SalesOrderID, OrderDate, DATEPART(yyyy,OrderDate) AS OrderYear,

DATENAME(m,OrderDate) AS OrderMonth FROM Sales.SalesOrderHeader;

1. **Write a query using the Sales.SalesOrderHeader table that displays the SubTotal rounded to two decimal places. Include the SalesOrderID column in the results.**
   1. SELECT SalesOrderID, ROUND(SubTotal,2) AS SubTotal FROM Sales.SalesOrderHeader;
2. **Modify the previous query so that the SubTotal is rounded to the nearest dollar but still displays two zeros to the right of the decimal place**
   1. SELECT SalesOrderID, ROUND(SubTotal,0) AS SubTotal FROM Sales.SalesOrderHeader;
3. **Write a statement that generates a random number between 1 and 10 each time it is run.**
   1. SELECT SQRT(SalesOrderID) AS OrderSQRT FROM Sales.SalesOrderHeader;
4. **Write a query using the HumanResources.Employee table to display the BusinessEntityID column.Display “Even” when the BusinessEntityID value is an even number or “Odd” when it is odd.**
   1. SELECT BusinessEntityID,

CASE BusinessEntityID % 2 WHEN 0 THEN 'Even' ELSE 'Odd' END

FROM HumanResources.Employee;

1. **Write a query using the Sales.SalesOrderDetail table to display a value (“Under 10” or “10–19” or “20–29” or “30–39” or “40 and over”) based on the OrderQty value. Include the SalesOrderID and OrderQty columns in the results.**
   1. SELECT SalesOrderID, OrderQty,

CASE WHEN OrderQty BETWEEN 0 AND 9 THEN 'Under 10'

WHEN OrderQty BETWEEN 10 AND 19 THEN '10-19'

WHEN OrderQty BETWEEN 20 AND 29 THEN '20-29'

WHEN OrderQty BETWEEN 30 AND 39 THEN '30-39'

ELSE '40 and over' end AS range

FROM Sales.SalesOrderDetail;

1. **Write a query using the Sales.SalesOrderHeader table to display the orders placed during 2001 by using a function. Include the SalesOrderID and OrderDate columns in the results.**
   1. SELECT SalesOrderID, OrderDate FROM Sales.SalesOrderHeader WHERE YEAR(OrderDate) = 2001;
2. **Write a query using the Sales.SalesOrderHeader table listing the sales in order of the month the order was placed and then the year the order was placed. Include the SalesOrderID and OrderDatecolumns in the results.**
   1. SELECT SalesOrderID, OrderDate FROM Sales.SalesOrderHeader ORDER BY MONTH(OrderDate), YEAR(OrderDate);
3. **Write a query that displays the PersonType and the name columns from the Person.Person table. Sort the results so that rows with a PersonType of IN, SP, or SC sort by LastName. The other rows should sort by FirstName**
   1. SELECT PersonType, FirstName, MiddleName, LastName

FROM Person.Person

ORDER BY CASE WHEN PersonType IN ('IN','SP','SC') THEN LastName

ELSE FirstName END;