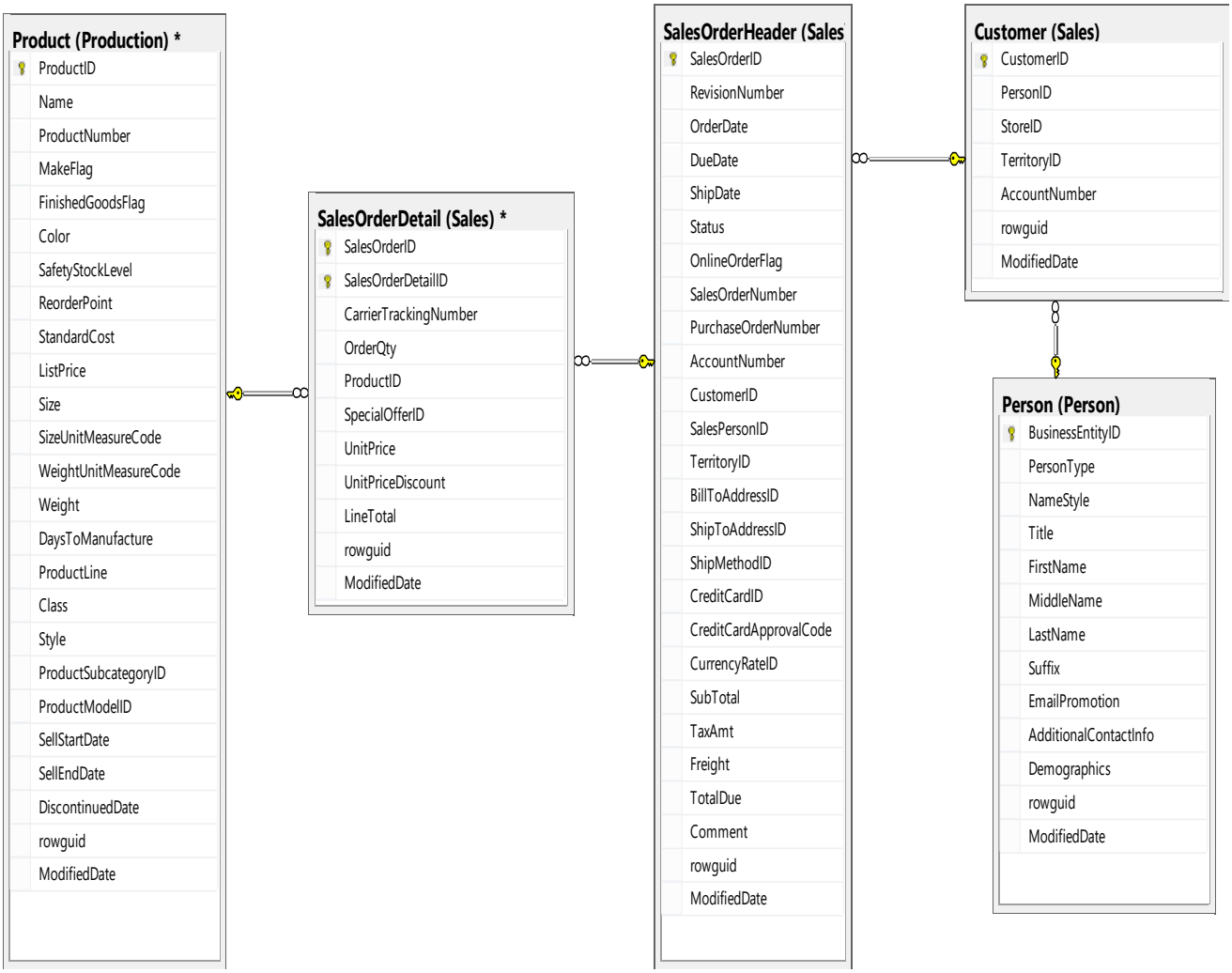
Lab 2 Exercises

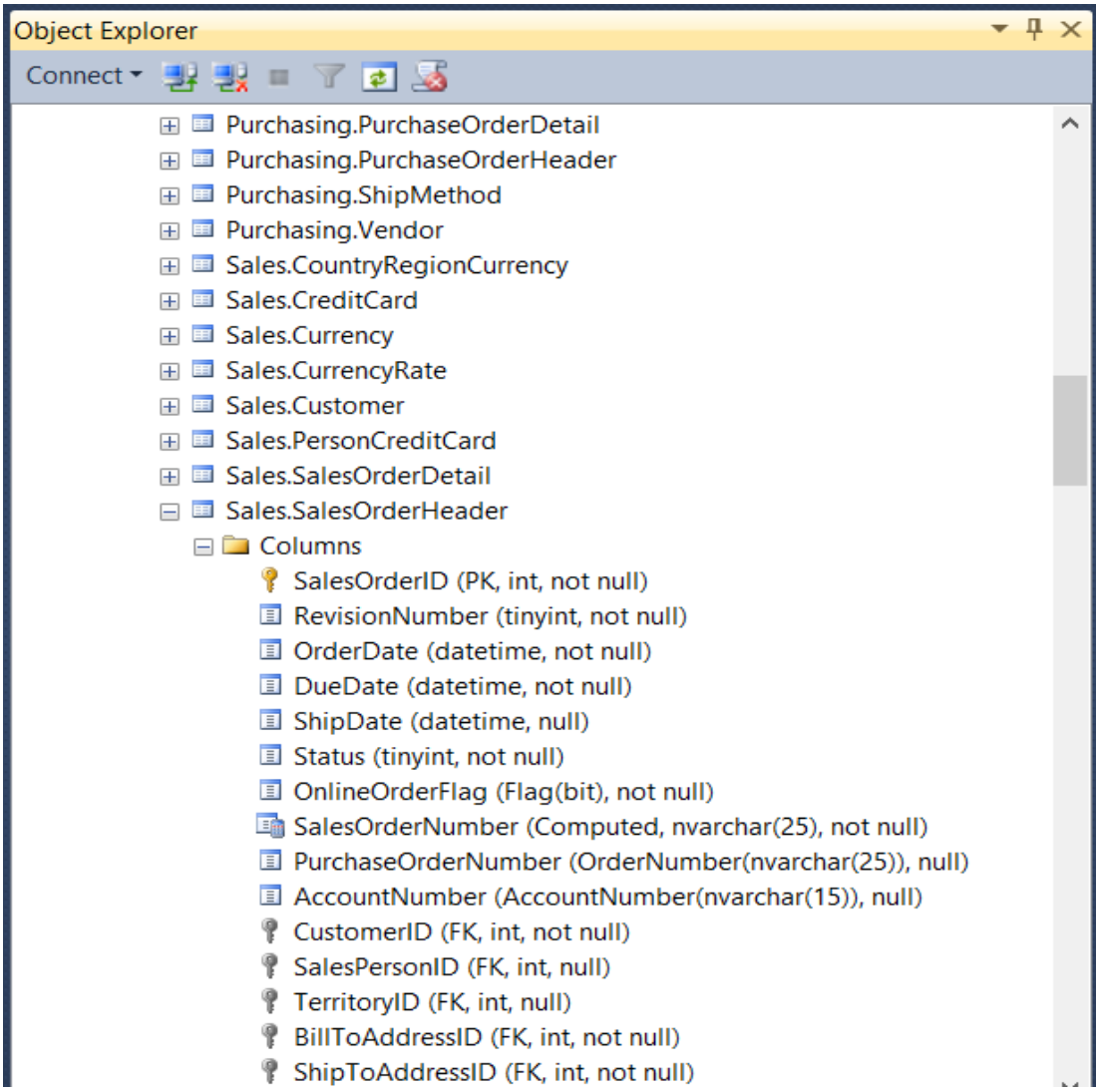
These exercise questions are for self-practice. No submission is needed.

Notes: The following partial ERD for AdventureWorks2008R2 was generated in SQL Server Management Studio. Use it to locate data when writing SQL queries.



Notes: If an ERD is not available, we can also use the Object Explorer in SQL Server Management Studio to locate data by following the steps listed below.

1) Under Object Explorer in SQL Server Management Studio, expand Databases
2) Expand the database we want to work with, such as AdventureWorks2008R2
3) Expand Tables
4) Expand the table we want to work with, such as Sales.SalesOrderHeader
5) Expand Columns
6) Then we'll see all columns contained in a table



```
USE AdventureWorks2008R2;


-- Exercise 1

/* Retrieve only the following columns from the
   Production.Product table:
                Product ID
                Name
                Selling start date
                Selling end date
                Size
                Weight */


-- Exercise 2

/* Select all info for all orders with no credit card id. */


-- Exercise 3

/* Select all info for all products with size specified. */


-- Exercise 4

/* Select all information for products that started selling
   between January 1, 2007 and December 31, 2007. */


-- Exercise 5

/* Select all info for all orders placed in June 2007 using date
   functions, and include a column for an estimated delivery date
   that is 7 days after the order date. */
```

```
-- Exercise 6

/* Determine the date that is 30 days from today and display only
   the date in mm/dd/yyyy format (4-digit year). */


-- Exercise 7

/* Determine the number of orders, overall total due,
   average of total due, amount of the smallest amount due, and
   amount of the largest amount due for all orders placed in May
   2008. Make sure all columns have a descriptive heading. */


-- Exercise 8

/* Using the 2007 data in the database, retrieve the Customer ID,
   total number of orders and overall total due for the customers
   who placed more than one order.

   Sort the result by the overall total due in the descending
   order. */


-- Exercise 9

/*
   Provide a unique list of the sales person ids who have sold
   the product id 777. Sort the list by the sales person id. */


-- Exercise 10

/* List the product ID, name, list price, size of products
   Under the ‘Bikes’ category (ProductCategoryID = 1) and
   Subcategory ‘Mountain Bikes’. */


-- Exercise 11

/* List the SalesOrderID and currency name for each order. */
```

Don’t look at the solution until you have completed an exercise question.

```
-- Solutions


USE AdventureWorks2008R2;


-- Exercise 1 Solution


/* Retrieve only the following columns from the
   Production.Product table:
                Product ID
                Name
                Selling start date
                Selling end date
                Size
                Weight
*/
SELECT ProductID, Name, SellStartDate, SellEndDate, Size, Weight
FROM Production.Product;


-- Exercise 2 Solution

/* Select all info for all orders with no credit card id. */
SELECT *
FROM Sales.SalesOrderHeader
WHERE CreditCardID IS NULL;



-- Exercise 3 Solution

-- Select all info for all products with size specified.
SELECT *
FROM Production.Product
WHERE Size IS NOT NULL;
```

```sql
-- Exercise 4 Solution

/* Select all information for products that started selling
   between January 1, 2007 and December 31, 2007. */

SELECT *
from Production.Product
WHERE SellStartDate BETWEEN '01/01/2007' AND '12/31/2007';


-- Exercise 5 Solution

/* Select all info for all orders placed in June 2007 using date
   functions, and include a column for an estimated delivery date
   that is 7 days after the order date. */

SELECT *, DATEADD(DAY, 7, OrderDate) AS [Est. Delivery Date]
FROM Sales.SalesOrderHeader
WHERE DATEPART(MONTH, OrderDate) = 6 and DATEPART(YEAR, OrderDate) =
2007;


-- Exercise 6 Solution

/* Determine the date that is 30 days from today and display only
   the date in mm/dd/yyyy format (4-digit year). */

SELECT CONVERT(CHAR(20), DATEADD(DAY, 30, GETDATE()), 101)
       AS [30 Days From Today];
```

```sql
-- Exercise 7 Solution

/* Determine the number of orders, overall total due,
   average of total due, amount of the smallest amount due, and
   amount of the largest amount due for all orders placed in May
   2008. Make sure all columns have a descriptive heading. */

SELECT COUNT(*) [Total Orders],
       SUM(TotalDue) [Overall Total],
       AVG(TotalDue) [Order Ave.],
       MIN(TotalDue) [Smallest Total],
       MAX(TotalDue) [Largest Total]
FROM Sales.SalesOrderHeader
WHERE (DATEPART(MONTH, OrderDate) = 5) AND
      (DATEPART(YEAR, OrderDate) = 2008);



-- Exercise 8 Solution

/* Retrieve the Customer ID, total number of orders and overall total
   due for the customers who placed more than one order in 2007
   and sort the result by the overall total due in the descending
   order. */

SELECT CustomerID,
       COUNT(SalesOrderID) [Order Count],
       SUM(TotalDue) [Overall Total Due]
FROM Sales.SalesOrderHeader
WHERE DATEPART(YEAR, OrderDate) = 2007
GROUP BY CustomerID
HAVING COUNT(SalesOrderID) > 1
ORDER BY [Overall Total Due] DESC;
```

```sql
-- Exercise 9 Solution

/*
   Provide a unique list of the sales person ids who have sold
   the product id 777. Sort the list by the sales person id.
*/


SELECT DISTINCT SalesPersonID
FROM Sales.SalesOrderHeader oh
INNER JOIN Sales.SalesOrderDetail od
ON oh.SalesOrderID = od.SalesOrderID
WHERE ProductID = 777
ORDER BY SalesPersonID;



-- Exercise 10

/* List the product ID, name, list price, size of products
   Under the ‘Bikes’ category (ProductCategoryID = 1) and
   Subcategory ‘Mountain Bikes’. */


select pdt.ProductID, pdt.Name, pdt.ListPrice, pdt.Size
from Production.Product pdt
join Production.ProductSubcategory psc
on pdt.ProductSubcategoryID = psc.ProductSubcategoryID
where psc.ProductCategoryID = 1 and psc.Name = ‘Mountain Bikes’;



-- Exercise 11

/* List the SalesOrderID and currency name for each order. */

select soh.SalesOrderID, crc.Name
from Sales.SalesOrderHeader soh
join Sales.CurrencyRate cr
on soh.CurrencyRateID = cr.CurrencyRateID
join Sales.Currency crc
on cr.ToCurrencyCode = crc.CurrencyCode;
```