Laboratory work 3

LEARN BASIC SQL DATA MANIPULATION COMMANDS

Prepare to work

1. Move the database file **SK.mdb** created during the previous work into another folder (e.g., D:\ACC_LAB_3).

2. Run DBMS Access.

3. Open the database created earlier.

Follow the steps below

I. SELECT SQL command. Data processing using SELECT queries.

To implement following SELECT SQL queries open “Create” tab in the Access window.

1. Print a list of products delivered by the supplier 1 (Ivanov I.I. PE) for the contract 1.

Steps:

1) Open “Query Design” window and select “Contracts”, “Supplied”, and “Suppliers” tables (figure 3.1);

2) Add the following fields to the query (TableName.FieldName): Contracts.ContractNumber, Supplied.Product, Supplied.Amount, Supplied.PricePerItem, Suppliers.SupplierName, Suppliers.SupplierID (figure 3.1);

3) Set value 1 as the “Criteria” for the fields “ContractNumber” and “SupplierID” (figure 3.1);

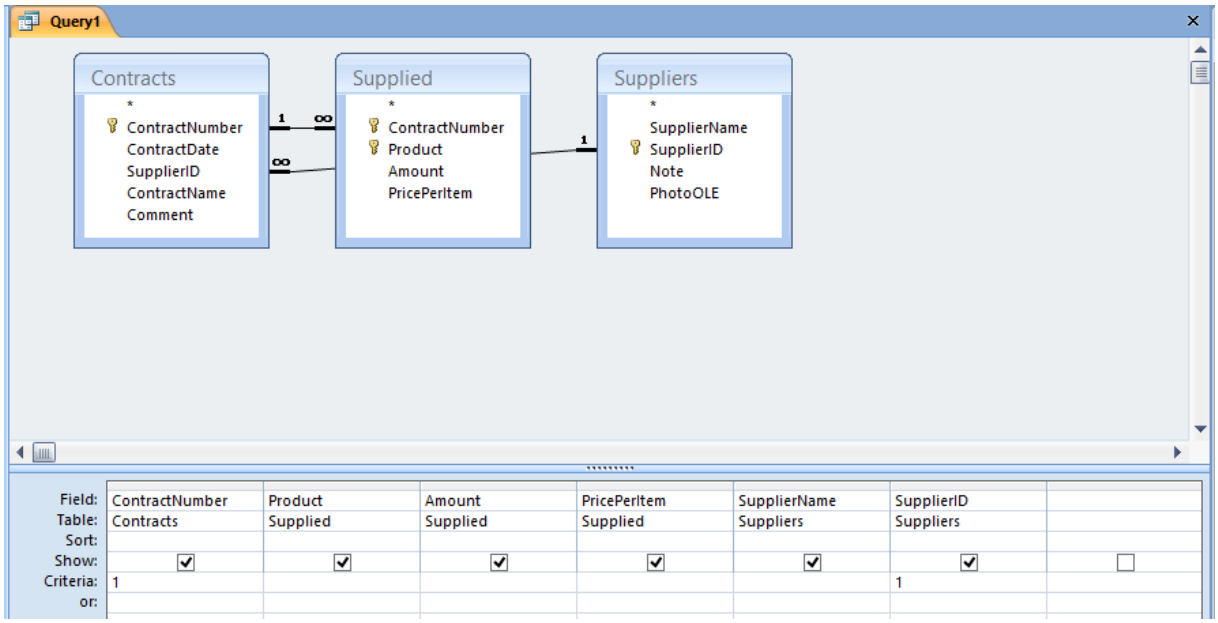

Figure 3.1

4) Switch “View” to “SQL View” and check SQL code of the SELECT command:

```
SELECT
      Contracts.ContractNumber,
      Supplied.Product,
      Supplied.Amount,
      Supplied.PricePerItem,
      Suppliers.SupplierName,
      Suppliers.SupplierID
FROM
      Suppliers INNER JOIN
            (Contracts INNER JOIN Supplied ON Contracts.ContractNumber =
            Supplied.ContractNumber)
            ON Suppliers.SupplierID = Contracts.SupplierID
WHERE
      (((Contracts.ContractNumber)=1) AND ((Suppliers.SupplierID)=1));
```

5) Switch “View” to “Datasheet View” and check result of query execution;

6) Save this query as “Query1” and close.

2. Print a list of products delivered by supplier 1 (Ivanov I.I. PE) between 9/1/1999 and 9/12/1999 (using "mm/dd/yyyy" date format).

Use the following SQL command to implement this query:

```
SELECT
      Supplied.ContractNumber,
      Contracts.ContractDate,
      Supplied.Product,
      Supplied.Amount,
      Supplied.PricePerItem,
      Suppliers.SupplierName
FROM
      Suppliers INNER JOIN
            (Contracts INNER JOIN Supplied ON Contracts.ContractNumber =
            Supplied.ContractNumber)
            ON Suppliers.SupplierID = Contracts.SupplierID
WHERE
      (((Contracts.ContractDate) Between #9/5/1999# And #9/12/1999#) AND
((Suppliers.SupplierID)=1));
```

You can use the query designer (figure 3.3) as well, but entering SQL commands is much more preferable and may prevent various mistakes.

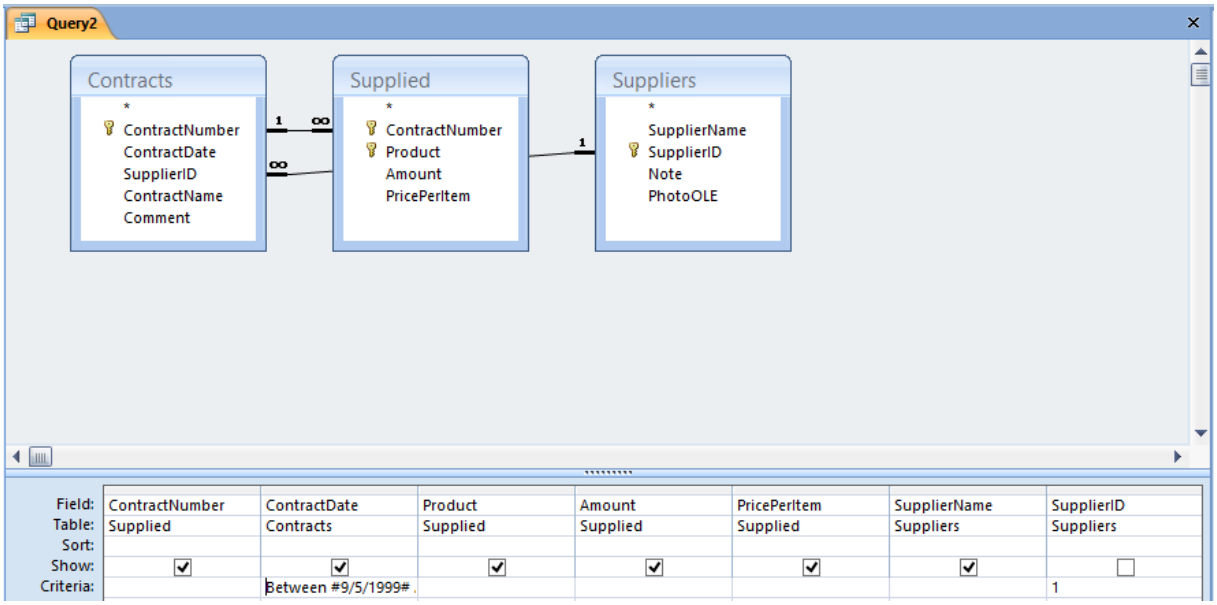

Figure 3.2

Save created query as “Query2” and close.

3. Print a list of products supplied in September of 1999 with supplier name and supply date.

Use the following SQL command or use the query designer (figure 3.3):

```
SELECT
      Contracts.ContractNumber,
      Contracts.ContractDate,
      Supplied.Product,
      Supplied.Amount,
      Supplied.PricePerItem,
      (Supplied.Amount * Supplied.PricePerItem) AS Total,
      Suppliers.SupplierName
FROM
      Suppliers INNER JOIN
            (Contracts INNER JOIN Supplied ON Contracts.ContractNumber =
            Supplied.ContractNumber)
            ON Suppliers.SupplierID = Contracts.SupplierID
WHERE
      Month(Contracts.ContractDate) = 9 AND Year(Contracts.ContractDate) =
1999
```

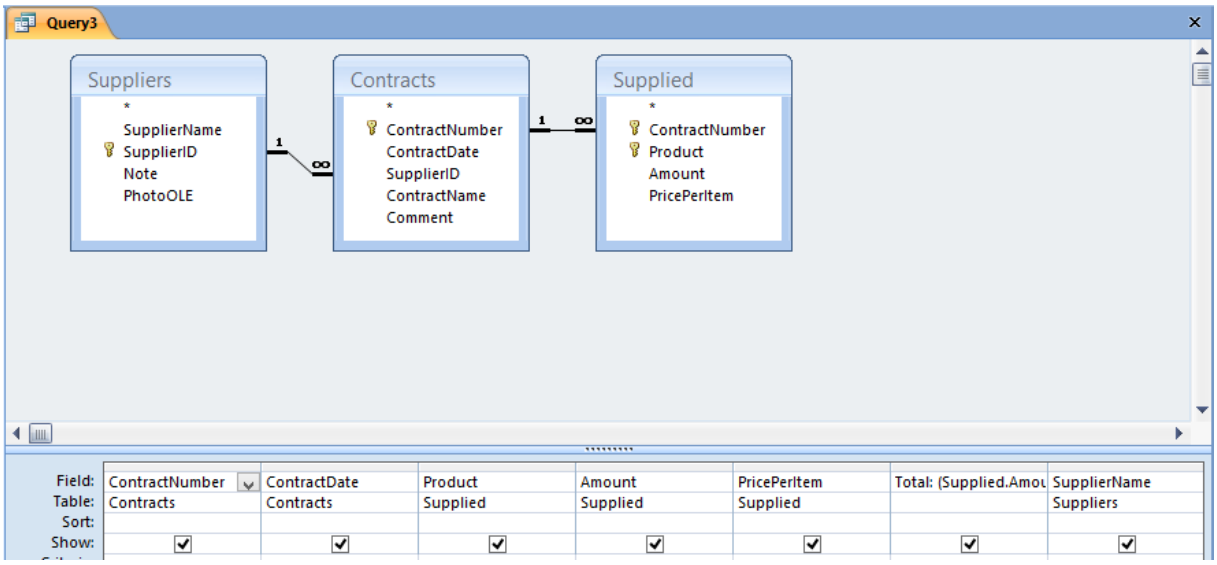

Figure 3.3

Save created query as “Query3” and close.

4. Print a list of contracts (number, date), total amount of the supplied products and total price for each contract (multiply and sum amount and price for each contract). The list should be sorted by contract numbers (ascending).

Use the following SQL command or use the query designer (figure 3.4):

```
SELECT
      Contracts.ContractNumber,
      Contracts.ContractDate,
      Sum(Supplied.Amount) AS [TotalAmount],
      Sum(Supplied.Amount * Supplied.PricePerItem) AS [TotalPrice]
FROM
      Contracts INNER JOIN
            Supplied ON Contracts.ContractNumber = Supplied.ContractNumber
GROUP BY Contracts.ContractNumber, Contracts.ContractDate
ORDER BY Contracts.ContractNumber
```

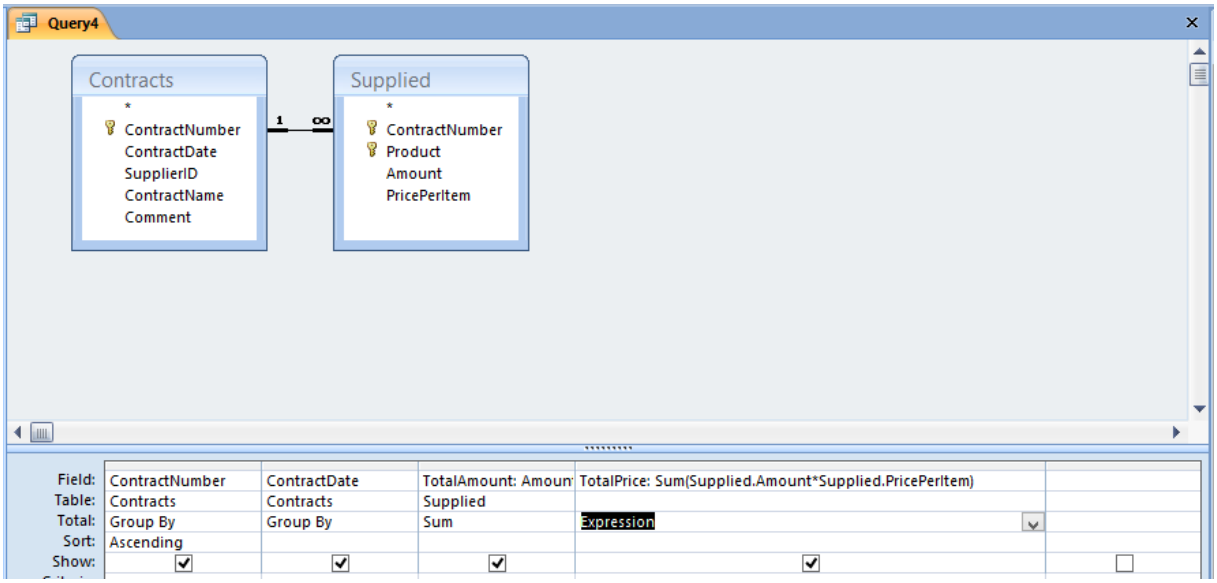

Figure 3.4

Save created query as “Query4” and close.

5. Print a list of contracts (number, date) with total price for each contract. The list should be sorted by total price for each contract. Records

for which contract number is greater than 3 should be excluded from query results.

Use the following SQL command or use the query designer (figure 3.5):

```
SELECT
      Contracts.ContractNumber,
      Contracts.ContractDate,
      Sum(Supplied.Amount * Supplied.PricePerItem) AS [TotalPrice]
FROM
      Contracts INNER JOIN
            Supplied ON Contracts.ContractNumber = Supplied.ContractNumber
GROUP BY Contracts.ContractNumber, Contracts.ContractDate
HAVING Contracts.ContractNumber <= 3
ORDER BY Sum(Supplied.Amount * Supplied.PricePerItem)
```

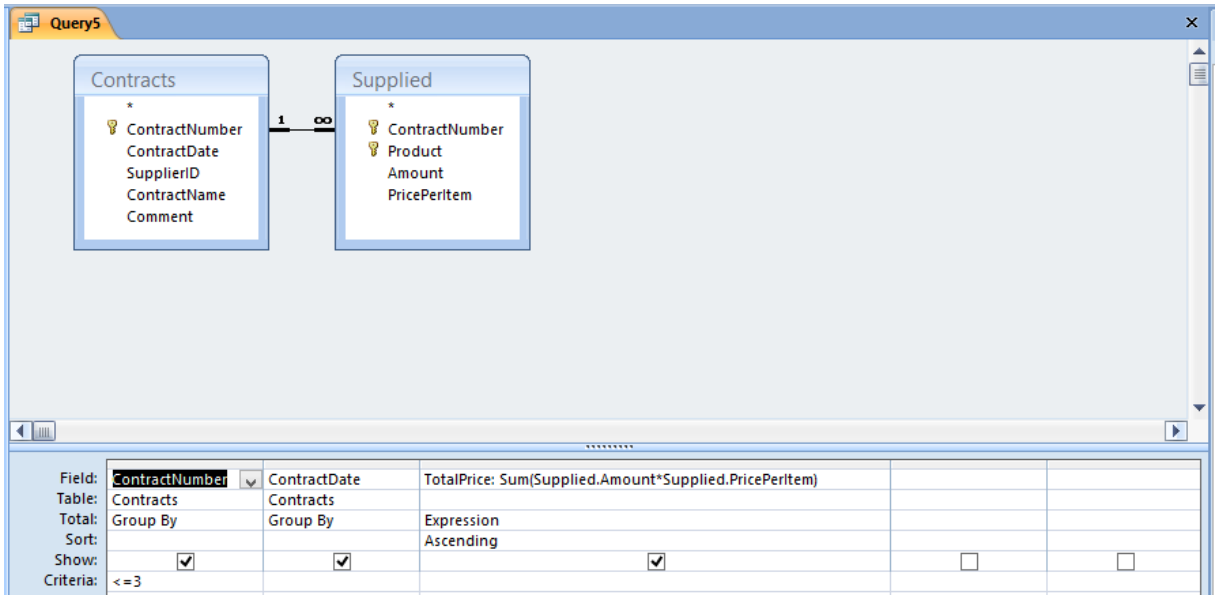

Figure 3.5

Save created query as “Query5” and close.

6. Print information about the largest product batch among all contracts. Include information about supplier, contract number, and date.

Use the following SQL command or use the query designer (figure 3.6):

```
SELECT
      Contracts.ContractNumber,
```

```
        Contracts.ContractDate,
        Supplied.Product,
        Supplied.Amount,
        Suppliers.SupplierName
FROM
        Suppliers INNER JOIN
              (Contracts INNER JOIN Supplied ON Contracts.ContractNumber =
              Supplied.ContractNumber)
              ON Suppliers.SupplierID = Contracts.SupplierID
WHERE
        Supplied.Amount = (SELECT Max(Amount) FROM Supplied)
```

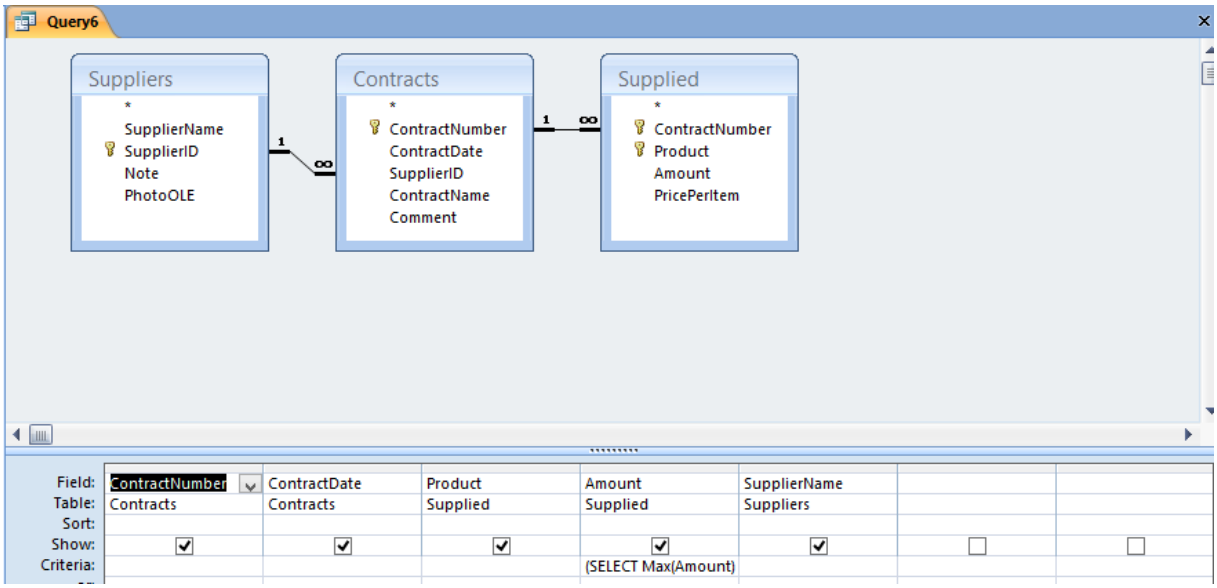

Figure 3.6

Save created query as “Query6” and close.

7. Print a list of suppliers (name and ID) that have not concluded any contracts.

Use the following SQL command or use the query designer (figure 3.7):

Option 1

```
SELECT
        Suppliers.SupplierName,
        Suppliers.SupplierID
FROM
```

```
        Suppliers
WHERE
        Suppliers.SupplierID NOT IN (SELECT SupplierID FROM Contracts)
```

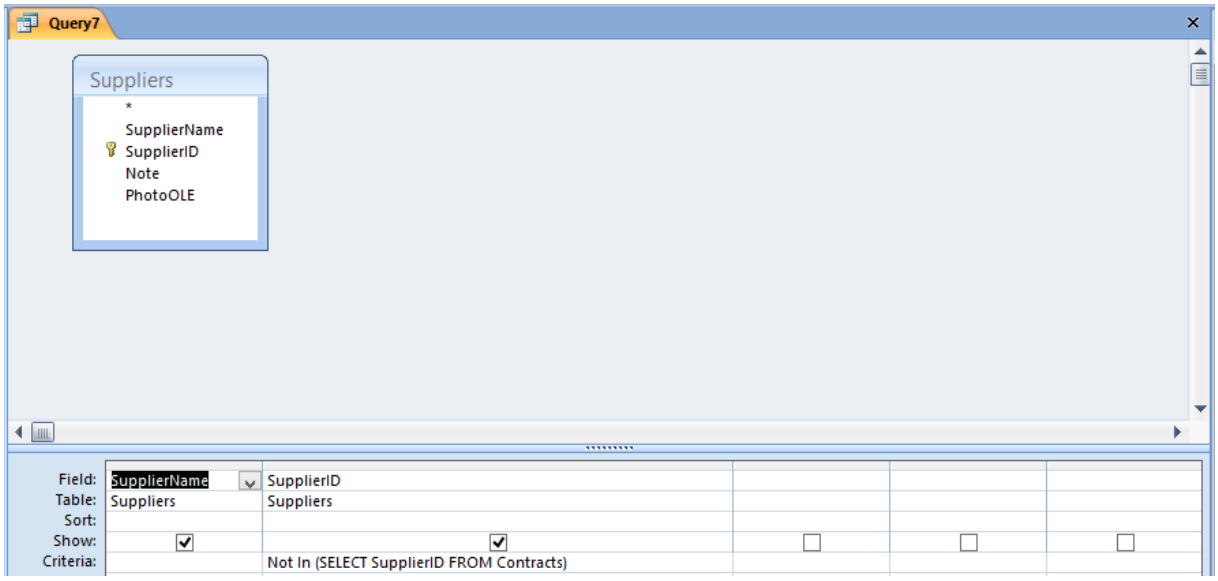

Figure 3.7a

Save created query as “Query7a” and close.

Option 2

```
SELECT
        Suppliers.SupplierName,
        Suppliers.SupplierID
FROM
        Suppliers
WHERE
        Suppliers.SupplierID <> ANY(SELECT SupplierID FROM Contracts)
```

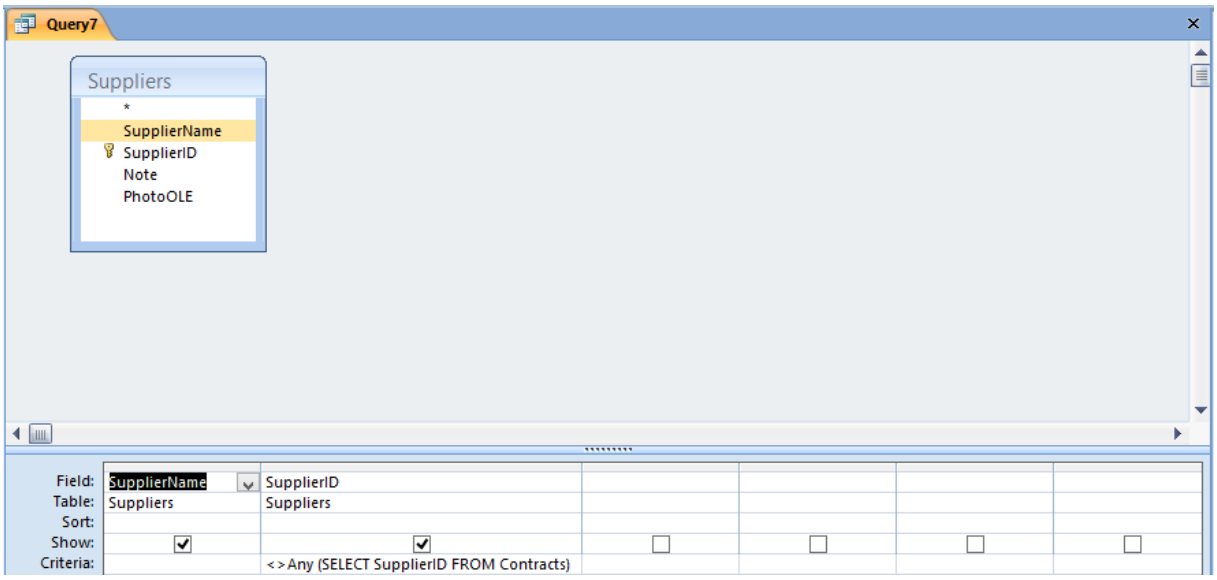

Figure 3.7b

Save created query as “Query7b” and close.

Warning! Both queries should give the same results.

But second option might give invalid result (e.g. print all suppliers) in various Access versions. To fix this issue modify the second query as following:

```
SELECT
      Suppliers.SupplierName,
      Suppliers.SupplierID
FROM
      Suppliers
WHERE
      NOT(Suppliers.SupplierID) = ANY(SELECT SupplierID FROM Contracts)
```

Ensure that both queries give the same results.

8. Print a list of supplied product names with the average price per item (regardless of supplier).

Use the following SQL command or use the query designer (figure 3.8):

```
SELECT
      Supplied.Product,
      Avg(Supplied.PricePerItem) AS [AvgPricePerItem]
FROM
```

```
      Supplied
GROUP BY Supplied.Product
```

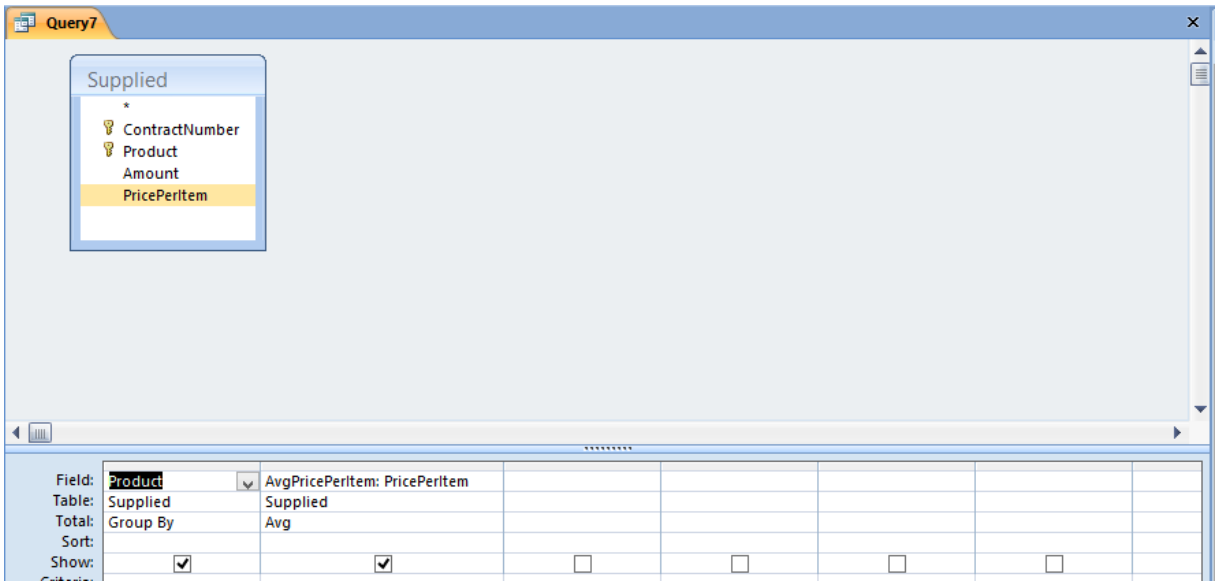

Figure 3.8

Save created query as “Query8” and close.

9. Print a list of products (name, amount and price, supplier) for which price per item is greater than average.

Use the following SQL command or use the query designer (figure 3.9):

```
SELECT
      Supplied.Product,
      Supplied.Amount,
      Supplied.PricePerItem,
      Suppliers.SupplierName
FROM
      Suppliers INNER JOIN
            (Contracts INNER JOIN Supplied ON Contracts.ContractNumber =
            Supplied.ContractNumber)
      ON Suppliers.SupplierID = Contracts.SupplierID
WHERE
      Supplied.PricePerItem > (SELECT Avg(PricePerItem) FROM Supplied)
```

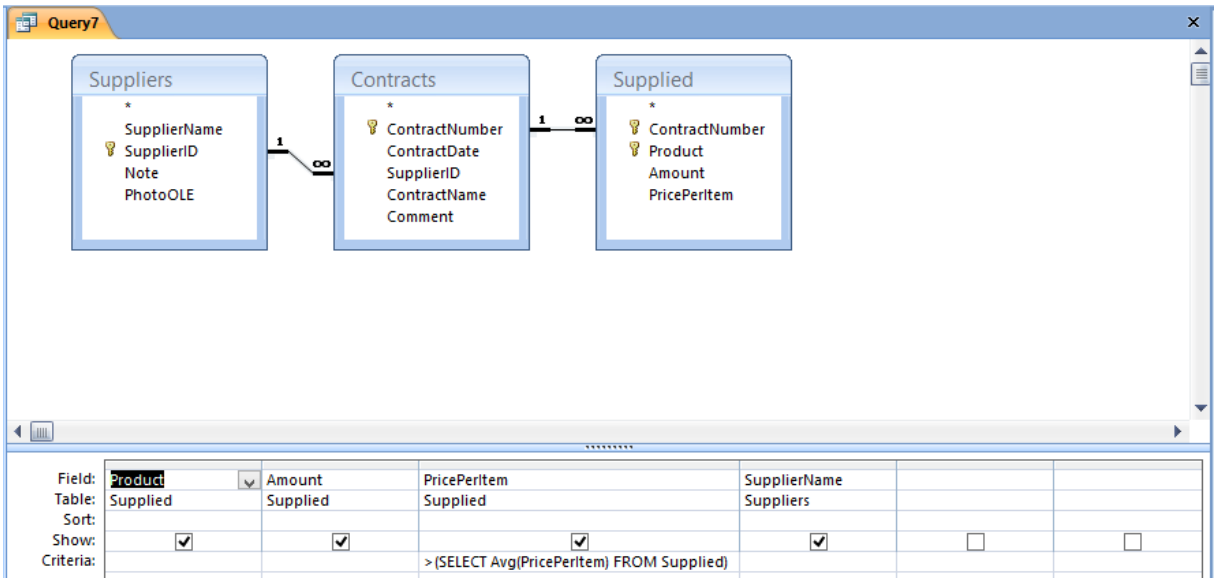

Figure 3.9

Save created query as “Query9” and close.

10. Print information about top five expensive products (name, price per item, supplier).

Use the following SQL command or use the query designer (figure 3.10):

```
SELECT TOP 5
      Supplied.Product,
      Supplied.PricePerItem,
      Suppliers.SupplierName
FROM
      (Suppliers INNER JOIN Contracts ON Suppliers.SupplierID = Con-
      tracts.SupplierID) INNER JOIN
      Supplied ON Contracts.ContractNumber = Supplied.ContractNumber
ORDER BY Supplied.PricePerItem DESC
```

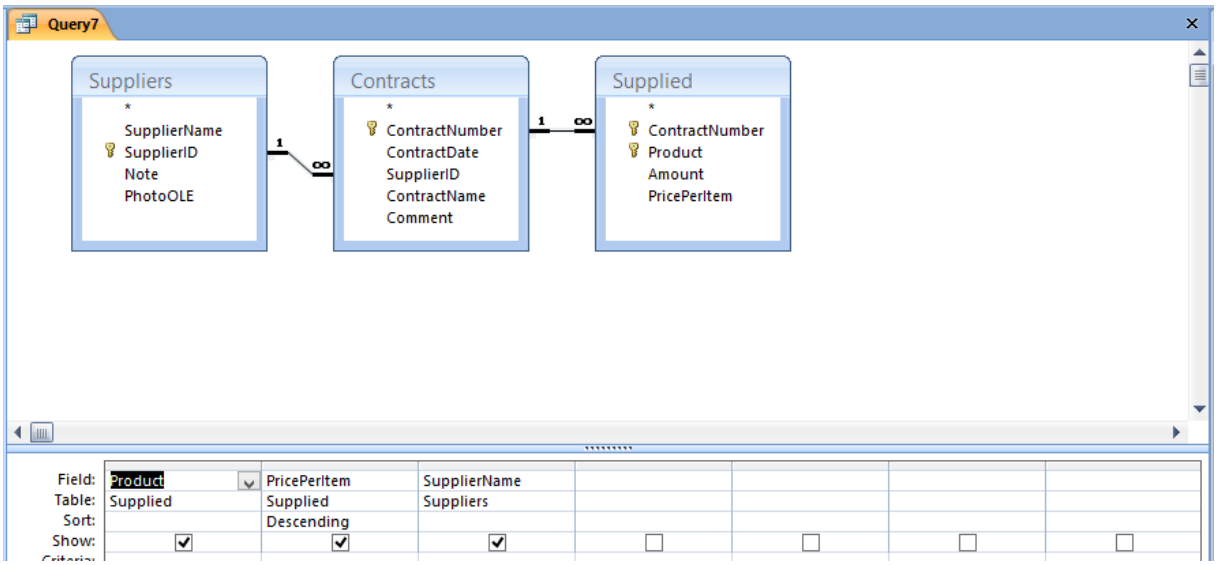

Figure 3.10

Save created query as “Query10” and close.

11. For each day of September 1999 define price of products delivered by each supplier (include only delivery days).

Use the following SQL command or use the query designer (figure 3.11):

```
TRANSFORM Sum(Supplied.Amount * Supplied.PricePerItem) AS [Total]
SELECT
      Suppliers.SupplierName
FROM
      Suppliers INNER JOIN
      (Contracts INNER JOIN Supplied ON Contracts.ContractNumber = Sup-
      plied.ContractNumber)
      ON Suppliers.SupplierID = Contracts.SupplierID
WHERE
      Month(Contracts.ContractDate) = 9 AND Year(Contracts.ContractDate) =
1999
GROUP BY
      Suppliers.SupplierName,
      Month(Contracts.ContractDate),
      Year(Contracts.ContractDate)
PIVOT Day(Contracts.ContractDate)
```

This is crosstab query, which defines [Suppliers.SupplierName] as row headings, Day(Contracts.ContractDate) as column headings, and [Total] as cell values.

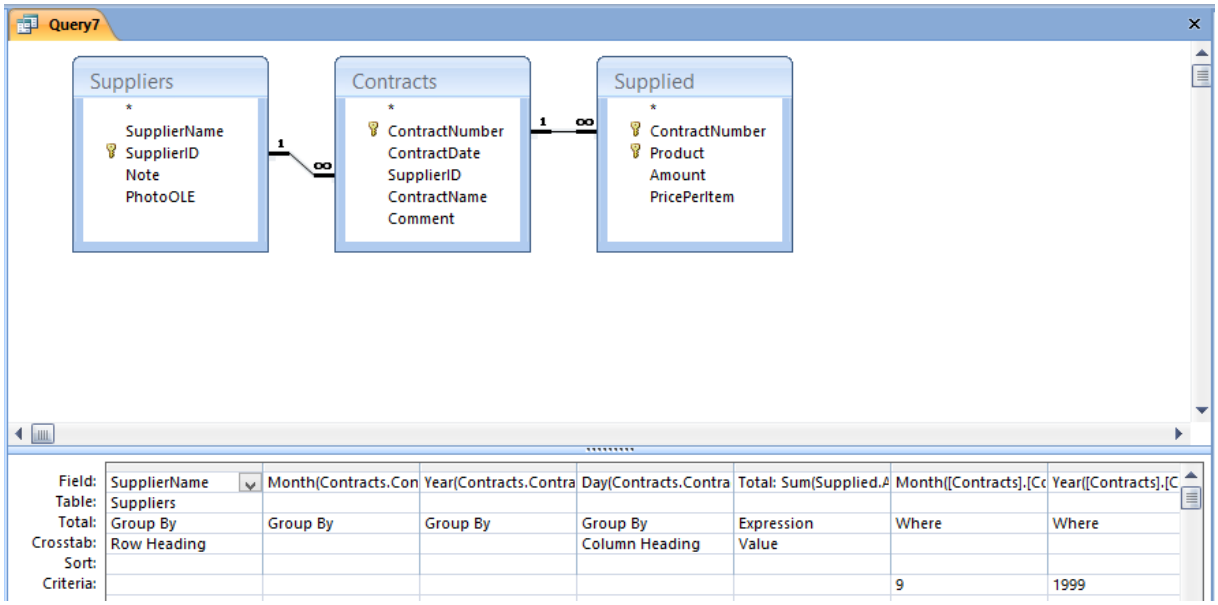

Figure 3.11

Save created query as “Query11” and close.

12. Create a list of contracts (only numbers), total amount of the supplied products, and total price for each contract. Print full names (last name, first name, and second name) of suppliers that are private entrepreneurs, as well as tax numbers of legal entities.

Use the following SQL command or use the query designer (figure 3.12):

```
SELECT
      Contracts.ContractNumber,
      Sum(Supplied.Amount) AS [TotalAmount],
      Sum(Supplied.Amount * Supplied.PricePerItem) AS [TotalPrice],
      LastName & " " & FirstName & " " & SecondName AS [SupplierFullName],
      TaxNumber
FROM
      ((Contracts LEFT JOIN IndividualEntrepreneurs ON Contracts.SupplierID
      = IndividualEntrepreneurs.SupplierID) LEFT JOIN
```

```
            LegalEntities          ON          Contracts.SupplierID          =
            LegalEntities.SupplierID) INNER JOIN
            Supplied ON Contracts.ContractNumber = Supplied.ContractNumber
GROUP BY
      Contracts.ContractNumber,
      LastName & " " & FirstName & " " & SecondName,
      TaxNumber
ORDER BY Contracts.ContractNumber
```

Here LEFT JOIN is used to join all records from “Contracts” and only matching records from “IndividualEntrepreneurs” and “LegalEntities” tables.

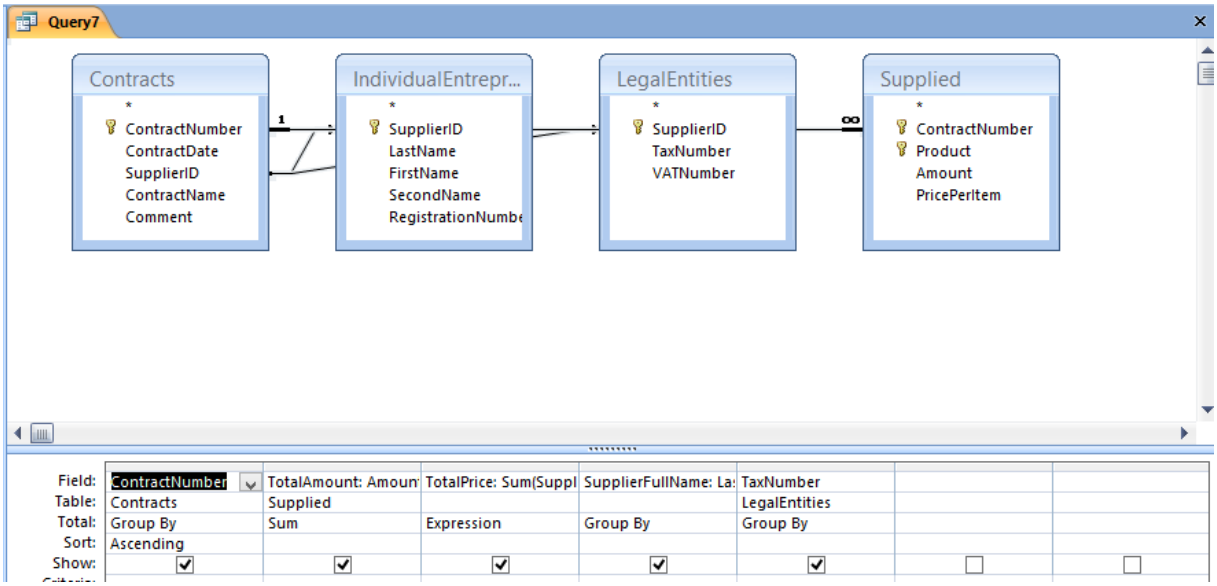

Figure 3.12

Save created query as “Query12” and close.

13. Define amounts of each delivered product by each supplier.

Use the following SQL command or use the query designer (figure 3.13):

```
TRANSFORM Sum(Supplied.Amount) AS [TotalAmount]
SELECT
      Suppliers.SupplierName
FROM
      Suppliers INNER JOIN
```

```
            (Contracts INNER JOIN Supplied ON Contracts.ContractNumber =
            Supplied.ContractNumber)
            ON Suppliers.SupplierID = Contracts.SupplierID
GROUP BY Suppliers.SupplierName
PIVOT Supplied.Product
```

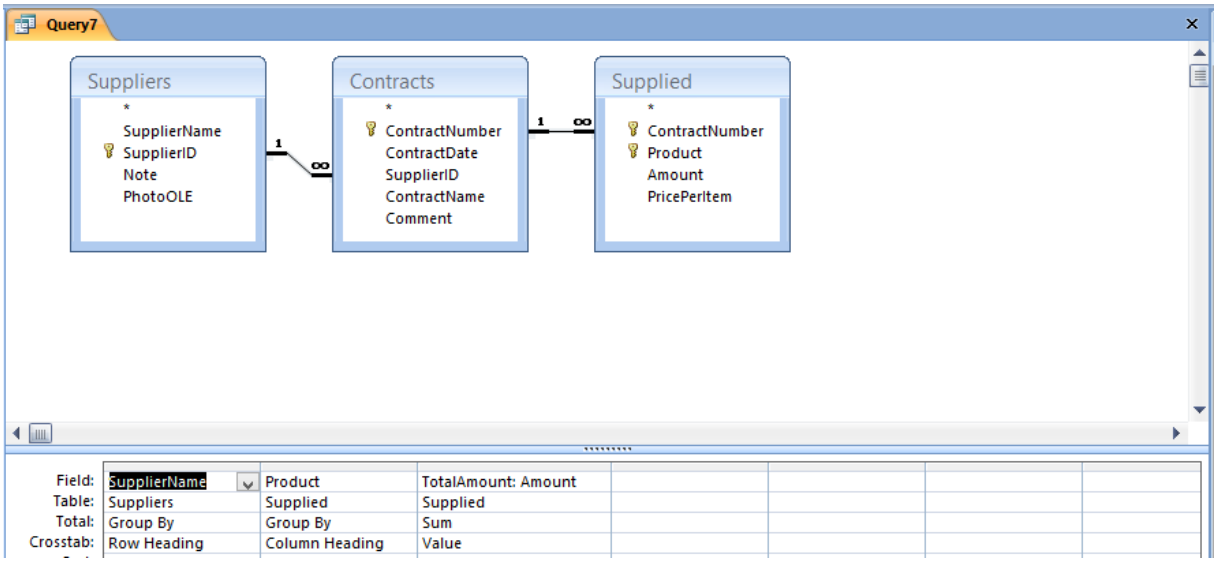

Figure 3.13

Save created query as “Query13” and close.

14. Print a list of contracts (number, date) and total price for each contract. The list should be sorted by total price for each contract. Exclude records for which the contract number is greater than a given value from the query result.

Use the following SQL command or use the query designer (figure 3.14):

```
PARAMETERS NumParam Short;
SELECT
      Contracts.ContractNumber,
      Contracts.ContractDate,
      Sum(Supplied.Amount) AS [TotalAmount],
      Sum(Supplied.Amount * Supplied.PricePerItem) AS [TotalPrice]
FROM
      Contracts INNER JOIN
      Supplied ON Contracts.ContractNumber = Supplied.ContractNumber
```

```
GROUP BY Contracts.ContractNumber, Contracts.ContractDate
HAVING Contracts.ContractNumber <= NumParam
ORDER BY Contracts.ContractNumber
```

This query is quite similar to the query 5. But in this query the contract number value that is used as criteria is not defined in the query. Instead the contract number is requested from a user when query is executes. This variable is called query parameter.

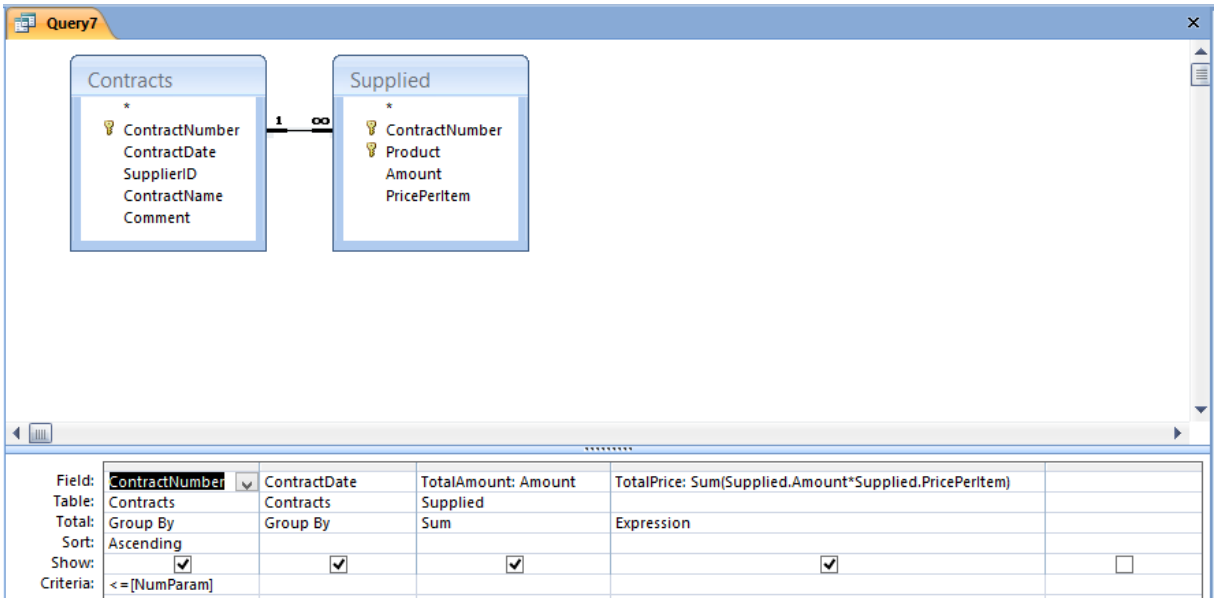

Figure 3.14

Save created query as “Query14” and close.

15. Create a list of products delivered by the suppliers 1 and 2 (“Interfruit” LLC).

Use the following SQL command (figure 3.15):

```
SELECT
      Product,
      Amount,
      PricePerItem,
      (Amount * PricePerItem) AS [TotalPrice]
FROM
```

```
      Contracts, Suppliers, Supplied
WHERE
      Contracts.ContractNumber = Supplied.ContractNumber AND
      Contracts.SupplierID = Suppliers.SupplierID AND
      Suppliers.SupplierID = 1
UNION
SELECT
      Product,
      Amount,
      PricePerItem,
      (Amount * PricePerItem) AS [TotalPrice]
FROM
      Contracts, Suppliers, Supplied
WHERE
      Contracts.ContractNumber = Supplied.ContractNumber AND
      Contracts.SupplierID = Suppliers.SupplierID AND
      Suppliers.SupplierID = 2
```

This query demonstrates usage of the UNION command that is used to merge results of several queries. It is not necessary to use UNION in order to solve considered task.

Figure 3.15

Save created query as “Query15” and close.

16. Create a list of products supplied more than once.

Use the following SQL command or use the query designer (figure 3.16):

Option 1

```
SELECT
	Supplied.Product,
	Count(Supplied.Product) AS [CountProducts]
FROM
	Supplied
GROUP BY Supplied.Product
HAVING Supplied.Product IN
	(SELECT
		Product
	FROM
		Supplied
	GROUP BY Product
	HAVING Count(Product) > 1)
```

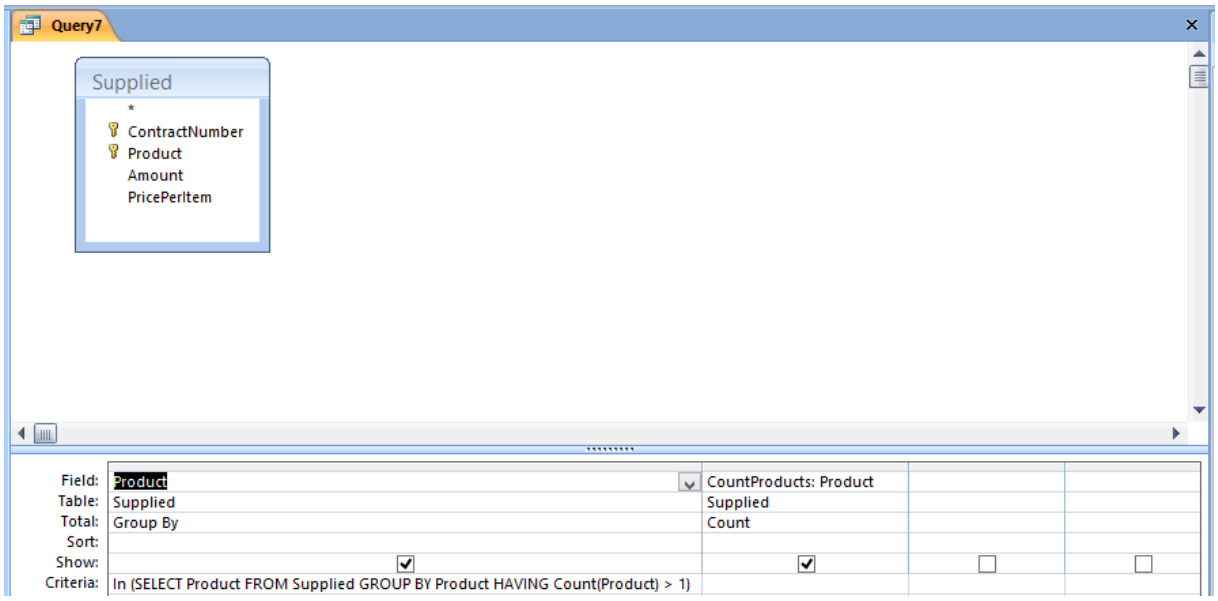

Figure 3.16a

Save created query as “Query16a” and close.

Option 2

```
SELECT
	Supplied.Product,
```

```
      Count(Supplied.Product) AS [CountProduct]
FROM
      Supplied
GROUP BY Supplied.Product
HAVING Count(Product) > 1
```

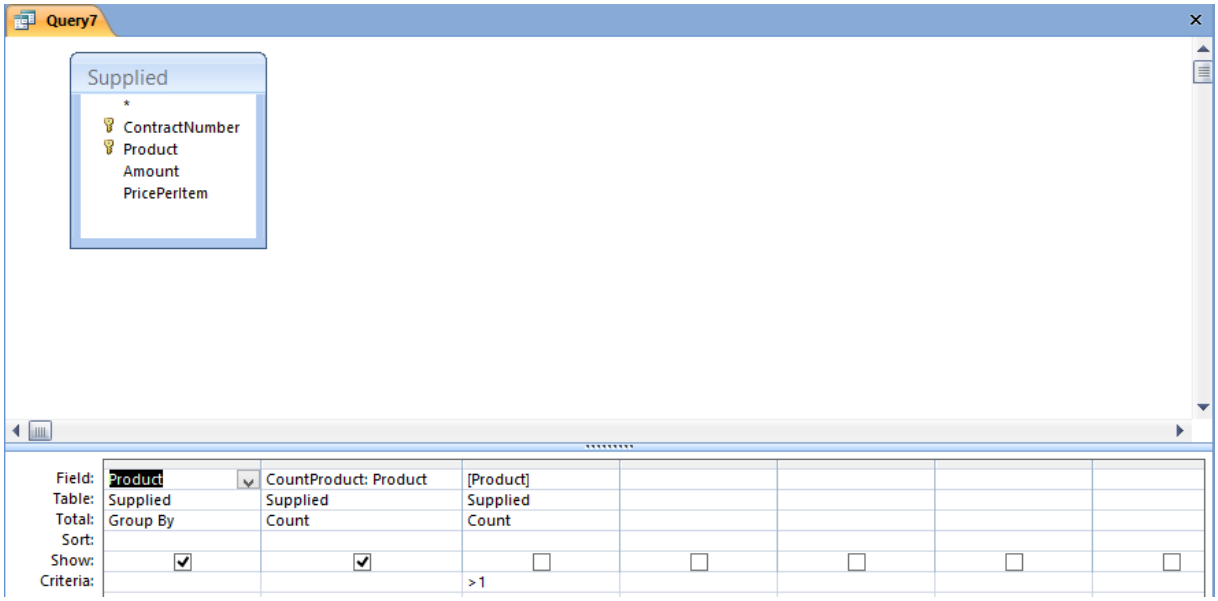

Figure 3.16b

Save created query as “Query16b” and close.

II. Data manipulation using SQL commands UPDATE and DELETE.

17. Increase amount of each product delivered by the supplier 1 by 10.

Use the following SQL command or use the query designer (figure 3.17):

```
UPDATE
      Supplied
SET
      Supplied.Amount = Supplied.Amount + 10
WHERE
Supplied.ContractNumber IN
      (SELECT
            ContractNumber
```

```
FROM
	Contracts
WHERE
	SupplierID = 1)
```

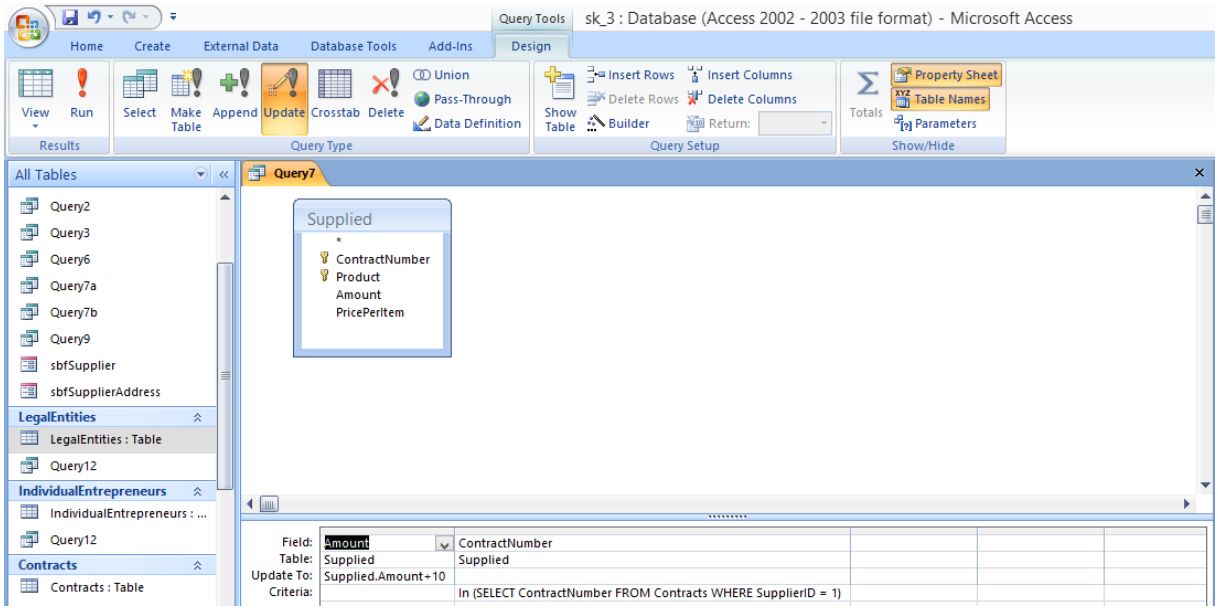

Figure 3.17

Save created query as “Query17” and close.

18. Delete all “empty” contracts (with no records about supplied products).

1) Create new record in “Contracts” table:

ContractNumber 8

ContractDate 7/27/2002

SupplierID 3

2) Use the following SQL command or use the query designer (figure 3.18):

```
DELETE
	Contracts.ContractNumber
FROM
	Contracts
WHERE
```

```
Contracts.ContractNumber NOT IN (SELECT ContractNumber FROM Supplied)
```

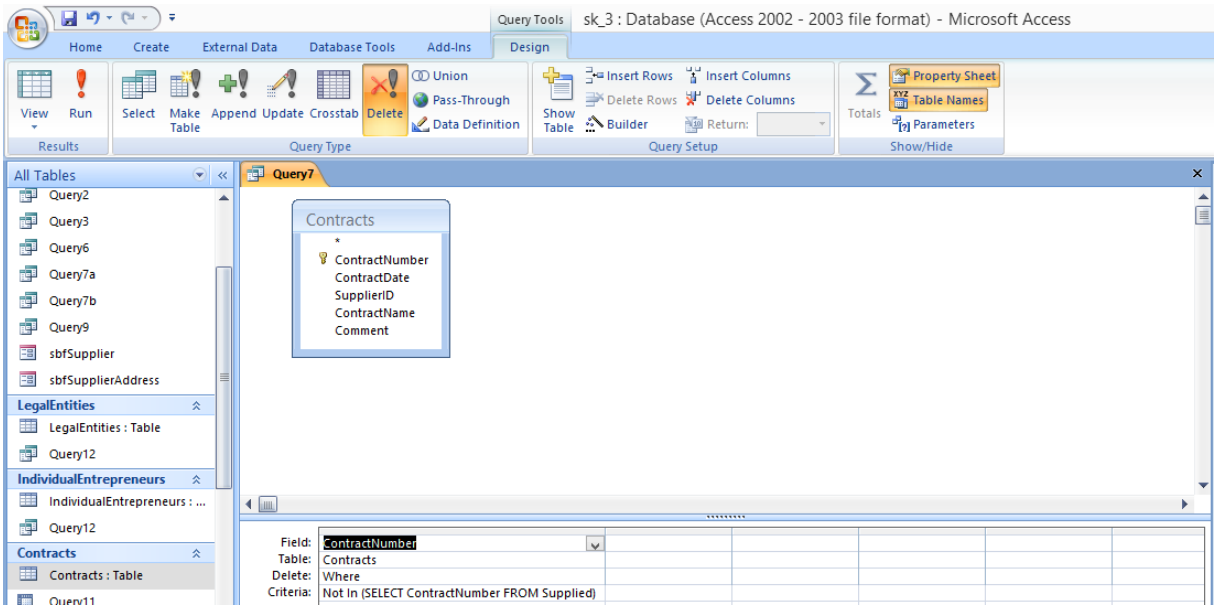

Figure 3.18

Save created query as “Query18” and close.

III. Finish the work. Save SK.mdb file.

Report requirements:

1) Briefly describe main steps of this work;

2) Depict query code and execution result for each SQL command.

Questions

1. SELECT SQL command. Description, purpose, and usage.
2. SELECT SQL command. Provided relational operations.
3. SELECT SQL command. Columns in query results. Expressions.
4. SELECT SQL command. FROM statement. Purpose and usage.
5. SELECT SQL command. WHERE statement. Purpose and usage.

6. SELECT SQL command. WHERE statement. FILTER conditions and their usage.

7. SELECT SQL command. WHERE statement. Multiple tables queries. JOIN conditions and their usage.

8. SELECT SQL command. Join of tables in multiple tables queries. INNER JOIN, LEFT JOIN, RIGHT JOIN and their features.

9. SELECT SQL command. Crosstab queries and their features.

10. SELECT SQL command. DISTINCT argument. Purpose and usage.

11. SELECT SQL command. TOP argument. Purpose and usage.

12. SELECT SQL command. Boolean operators AND, OR, NOT, and their usage.

13. SELECT SQL command. Special operators IN, BETWEEN. Purpose and usage.

14. SELECT SQL command. Aggregate functions COUNT, SUM, AVG, MAX, MIN. Purpose and usage.

15. SELECT SQL command. GROUP BY statement. Purpose and usage.

16. SELECT SQL command. ORDER BY statement. Purpose and usage.

17. SELECT SQL command. HAVING statement. Purpose and usage.

18. SELECT SQL command. Subqueries. Purpose and usage.

19. SELECT SQL command. Subqueries. Types and features.

20. SELECT SQL command. Query parameters and their usage.

21. INSERT SQL command. Description, purpose, and usage.

22. DELETE SQL command. Description, purpose, and usage.

23. UPDATE SQL command. Description, purpose, and usage.

24. How to implement query 15 without UNION operation?