SQL – Assignments

Chapter 2:

1. Connect to the sql server and use the GlobalToyz database
2. Display all details of all toys
3. Display the name, address, and the zip code of all recipients in the following format:

Recipient First name	Recipient Last name	Address	Zip

4. Display the first name and the last name of all shoppers who live in the state of California
5. Display all details of orders that have a total cost exceeding $75
6. Display the details of the toys that belong to the 'Largo' brand
   Hint: The brand ID for Largo is 004
7. Display the orders placed on May 22, 2001.
8. Display a report for all orders in the following format:

Order Number	Shipping Charges	Gift Wrap Charges	Handling Charges

9. Display a list of all toys in the price range of $10 to $20
10. Display the first name, the last name and the email ID of shoppers who belong to the states of California and Illinois
11. Display the orders placed on May 20th, 2001 that have a total cost of more than $75 in the following format:

Order Number	Order Date	Shopper ID	Total Cost

12. Display details of orders that do not have any messages attached.
13. Display names and prices of all toys in the following format. Ensure that the toy with the highest price is at the top of the list.

Toy Name	Toy Rate

14. Display names of toys and the lower age limit for the toys that cost less than $20, in ascending order.
15. Display the order number, the shopper ID, and the total value of the order in ascending order of the total value.
16. Display the total number of toys being sold at Global Toyz
17. Display the maximum, minimum, and the average rate of toys.
18. Display the total value of all orders put together.

19. Display the toy IDs of the top five 'Pick of the Month' toys based on the number of toys sold in the year 2000
20. Display all details of toys that have the word 'Racer' in the toy name
21. Display all shoppers whose names begin with 'S'.
22. Display the order number, the toy ID, and the description of gift wrappers used for all toys in the following format:

Order Number	Toy ID	Wrapper Description

23. What will be the output of the following code written for displaying the total order value for each order
    SELECT corderNo, ctoyId, sum(mToyCost) FROM OrderDetail GROUP BY cOrderNo
24. You can place an order for more than one toy. Display a report containing the order number and the total cost of toys for those orders whose total cost of toys is more than $50.
25. The following two codes containing the COMPUTE BY clauses generate errors. What are the possible causes of such errors:
    SELECT cToyID, mToyCost FROM OrderDetail
    COMPUTE SUM(mToyCost) BY cToyId

    SELECT cCartId FROM Orders
    ORDER BY cCartId
    COMPUTE AVG(mTotalCost) BY cCartId

Chapter 3:

1. Display the name, the description, and the rate for all toys. However, only the first 40 letters of the description should be displayed.
2. Display a report containing all the shipments in the following format:

Order Number	Shipment Date	Actual Delivery Date	Days in transit

Hint: Days in transit = Actual Delivery date – Shipment date

3. Display a report for the order number 000009 in the following format:

Order Number	Days in Transit

4. Display all orders in the following format:

Order Number	Shopper ID	Day of Order	Week day

5. Display names of all toys and categories to which they belong.
6. Display the names, brands and categories of all toys in the following format:

Toy Name	Brand	Category

7. Display the order number, the toy ID, and the description of gift wrappers used for all toys in the following format:

Order Number	Toy ID	Wrapper Description

8. Display the name of toys and the shopping cart ID for all toys. If the toys are not on the shopping cart, NULL should be displayed.

Toy Name	Cart ID
Robby the Whale	000005
Water Channel System	NULL

Hint: use left outer join.

9. Display the names of all the shoppers along with their initials in the following format:

Initials	vFirstName	vLastName

10. Display the Order number, order date and the quarter in which each of the orders where placed in the following format:

cOrderNo	dOrderDate	Quarter
000001	2001-05-20 00:00:00.000	2

11. Display the average price of all the toys rounded off to a whole number.
12. Display all the ShopperIDs along with the corresponding order number (if applicable) of the recipient who are located in the same state:
    Hint: Use outer join.

Chapter 5

1. Copy details of all toys priced more than $20 into a new table called PremiumToys
2. Display the first names, last names, addresses, and cities of shoppers and recipients in the following format:

First Name	Last Name	Address	City

3. Display the names of the toys that belong to the category 'Stuffed Toys'
4. Create a recipient table to store details of recipients. The following details about a recipient are to be stored in that table:

Attribute Name	**Data**
Order Number	000035
First Name	Shirley
Last Name	Nelson
Address	56700 Chain Boulevard Apartment #899
City	Austin
State	Texas
Country Code	001
Zip Code	78728
Phone	409-2387

5. The Recipient table and the Country table do not have the same data type for the cCountryID attribute. The sample structure of the two tables is shown below:

RECIPIENT	
Attribute Name	**Datatype**
cOrderNo	Char(6)
vFirstName	Varchar(20)
vLastName	Varchar(20)
vAddress	Varchar(20)
cCity	Char(15)

cState	Char(15)
cCountryId	Char(3)
cZipCode	Char(10)
cPhone	Char(15)

COUNTRY	
Attribute Name	**Datatype**
cCountryId	Char(3)
cCountry	Char(25)

Recreate the Recipient and the Country table so that they have the same datatype for the cCountryId attribute

6. Delete the Recipient table
7. Consider the following table structures:

TOYS	
Attribute Name	**Datatype**
cToyID	Char(6)
cToyName	Varchar(20)
vToyDescription	Varchar(250)
cCategoryID	Char(3)
mToyRate	Money
cBrandId	Char(3)
imPhoto	Image
siToyQoH	smallint
siLowerAge	Smallint
siUpperAge	Smallint
siToyWeight	Smallint
vToyImgPath	Varchar(50)

CATEGORY	
Attribute Name	**Datatype**
cCategoryID	Char(3)
cCategory	Char(20)
vDescription	Varchar(250)

ToyBrand	
Attribute Name	**Datatype**
cBrandId	Char(3)
cBrandName	Char(20)

Refer to these tables structures for the following problems:

a. Create the Category table. Enforce the following data integrity rules while creating the table
   i. The category id should be the primary key
   ii. The cCategory attribute should be unique but not the primary key.
   iii. The description of the categories attribute can allow storage of NULL values
b. Create the ToyBrand table. Enforce the following data integrity rules while creating the table
   i. The brand id should be the primary key
   ii. The brand name should be unique, but not the primary key
c. Create the Toys table with the following data integrity rules:
   i. The toy id should be the primary key
   ii. The quantity on hand (siToyQoH) of toys should be between 0 and 200
   iii. The imPhoto, vToyImgPath attributes can allow storage of NULL values
   iv. The cToyName, vToyDescription should not allow NULL values
   v. The lower age for the toys should be 1 by default
   vi. The values of the cCategoryId attributes should be present in the Category table
d. Modify the Toys table to enforce the following data integrity rules
   i. The values entered in the cBrandId attribute should be present in the ToyBrand table
   ii. The upper age for the toys should be 1 by default

8. The following code was used to copy details of all Shoppers from the state of California into a new table called CaliforniaShoppers. When the code was executed, it generated an error. Identify the error and rectify it

```
SELECT * FROM Shopper INTO CaliforniaShoppers
WHERE cState = 'California'
```

9. The following code was used to remove the table called CaliforniaShoppers

DELETE TABLE CaliforniaShoppers

The code generates an error and aborts. Identify the error and rectify it.

10. Display the first name and the last name of all the shoppers living in the cities Woodbridge, 'San Jose' or Las Vegas.
11. Display the names of all toys for whom no wrapper was used.
12. Display the order numbers for orders that has been already shipped.

Chapter 7:

1. Consider the following table structures.

TOYS	
Attribute Name	**Datatype**
cToyID	Char(6)
cToyName	Varchar(20)
vToyDescription	Varchar(250)
cCategoryID	Char(3)
mToyRate	Money
cBrandId	Char(3)
imPhoto	Image
siToyQoH	smallint
siLowerAge	Smallint
siUpperAge	Smallint
siToyWeight	Smallint
vToyImgPath	Varchar(50)

CATEGORY	
Attribute Name	**Datatype**
cCategoryID	Char(3)
cCategory	Char(20)
vDescription	Varchar(250)

ToyBrand	
Attribute Name	**Datatype**
cBrandId	Char(3)
cBrandName	Char(20)

Refer to these table structures for the following problems:

a. Create the Category table
b. Create the toyBrand table
c. Create the Toys table
d. Without modifying the Toys table that you have created, implement the following data integrity rules:
    i. The price of the toys should be greater than zero
    ii. The weight of the toys should be 1 by default
e. Store the following brands in the database

Brand Id	**Brand Name**
001	Bobby
002	Frances-Price
003	The Bernie Kids
004	Largo

f. Store the following categories of toys in the database

Category Id	**Category**	**Description**
001	Activity	Activity toys encourage the child's social skills and interest in the world around them
002	Dolls	A wide range of dolls from all the leading brands
003	Arts and Crafts	Encourage children to create masterpieces with these incredible craft kits.

g. Store the following details in the database

Attribute Name	**Datatype**
Toy ID	000001
Toy Name	Robby the Whale
Toy Description	A giant Blue Whale with two heavy duty handles that allow a child to ride on its back

Category ID	001
Toy Rate	8.99
Brand Id	001
Photo	NULL
Toy Quantity on Hand	50
Lower Age	3
Upper Age	9
Toy Weight	1
Toy Image Path	NULL

h. Increase the rate of the toy by $1 for the toy whose toy id is '000001'
i. Remove the 'Largo' brand from the database

2. The following query displays the first name and the total cost of the toy that a shopper has ordered:

```
SELECT vFirstName, mTotalCost
FROM Shopper JOIN Orders
On Shopper.cShopperId = Orders.cShopperId
```

3. The Toys table is being used heavily for querying. The queries are based on the cToyId attribute. You have to optimize the execution of queries. Also, ensure that the cToyId attribute is not duplicated.
4. The Category table is being used heavily for queries. The queries are based on the cCategoryId attribute of the table. The cCategoryId attribute is defined as the primary key. Create an appropriate index on the table so that the queries execute fast. Also, ensure that the cCategoryId attribute should not be duplicated.

5. Recently there have been complaints that all queries involving the Shopper table take longer to execute. You suspect that the table is fragmented. Use appropriate commands to verify the fragmentation.

6. The following code generates an error and stops executing:

```
DBCC SHOWCONFIG (Wrapper)
```

Detect the error and make appropriate changes in the code

7. Queries for displaying Recipient details pertaining to a state take a long time to execute. To enhance the performance of such queries you decide to incorporate a non-clustered index on the cState column of the recipient table by using the following code

   CREATE NONCLUSTERED INDEX idx_cState
   Recipient (cState)
   The above code generates an error and aborts. Detect the problem and rectify it.

8. The following code was used to add the details of a new shopping cart:
   INSERT TALBE ShoppingCart VALUES('000010', '000013',2)
   When the above statement was executed, it gave an error. Identify the problem in the above code.

Chapter 9:

1. Some of the queries to be performed are as follows:

   To display the names of Shoppers and the names of toys ordered by them

```
SELECT Shopper.vFirstName, vToyName
FROM Shopper JOIN Orders
ON Shopper.cShopperId = Orders.cShopperId
JOIN OrderDetail
ON Orders.cOrderNo = OrderDetail.cOrderNo
JOIN Toys
ON OrderDetail.cToyId = Toys.cToyId
```

   To display the names of shoppers, the names of toys ordered, and the quantity ordered

```
SELECT Shopper.vFirstName, vToyName, siQty
FROM Shopper JOIN Orders
ON Shopper.cShopperId = Orders.cShopperId
JOIN OrderDetail
ON Orders.cOrderNo = OrderDetail.cOrderNo
JOIN Toys
ON OrderDetail.cToyId = Toys.cToyId
```

   To display the names of shoppers, the names of toys ordered, and cost of the toys

```
SELECT Shopper.vFirstName, vToyName, mToyCost
FROM Shopper JOIN Orders
ON Shopper.cShopperId = Orders.cShopperId
JOIN OrderDetail
ON Orders.cOrderNo = OrderDetail.cOrderNo
JOIN Toys
ON OrderDetail.cToyId = Toys.cToyId
```

   Simplify the task of performing these queries.

2. A view has been defined as follows:

```
CREATE VIEW vwOrderWrapper
AS
SELECT cOrderNo, cToyId, siQty, vDescription, mWrapperRate
FROM OrderDetail JOIN Wrapper
ON OrderDetail.cWrapperId = Wrapper.cWrapperId
```

The following update command gives an error when the siQty and the mWrapperRate attributes are being updated using the following update command:

```
UPDATE  vwOrderWrapper
SET siQty = 2, mWrapperRate = mWrapperRate + 1
FROM vwOrderWrapper
WHERE cOrderNo = '000001'
```

Modify the update command to update the required values in the base table.

3. The status of shipment for the order '000003' is required. If the order has been delivered, then the message 'The order has been delivered', should be displayed, otherwise, the message 'The order has been shipped but not delivered' should be displayed.

   Hint: If the order has been shipped but not delivered, then the cDeliveryStatus attribute would contain 's' and if the order has not been delivered, the the cDeliveryStatus attribute would contain 'd'.

4. Increase the price of each toy by $0.5 until the average price of toys reaches near $22.5
5. Increase the price of each toy by $0.5 until the average price of toys reaches near $24.5. In addition, the maximum price of any toy should not exceed $53.
6. A report containing the name, the description, and the price for all the toys is required frequently. Create an object in the database so that there is no delay in getting the report due to network congestion.
7. The query for a report is as follows:
   ```
   SELECT vFirstName, vLastName, vEmailId
   FROM Shopper
   ```
8. Create a stored procedure that accepts a toy id and displays the name and price for that toy.
9. Create a stored procedure that would add the following data in the ToyBrand table

Brand Id	**Brand Name**
009	Fun world

10. Create a stored procedure called prcAddCategory that would add the following data in the Category table.

Category Id	**Category**	**Description**
018	Electronic Games	These games contain a screen with which children interact

11. Delete the procedure prcAddCategory
12. Create a procedure called prcCharges that returns the shipping charges and the wrapper charges for a given order number
13. Create a procedure called prcHandlingCharges that accepts an order number and displays handling charges. The prcHandlingCharges procedure should use the prcCharges procedure to get the shipping and the gift wrap charges
    Hint: Handling charges = Shipping charges + Gift wrap charges
14. The following code was used to create a procedure for displaying the respective brand name for a given brand id:

    ```
    CREATE PROCEDURE prcBrand @cBrandId char(3) OUTPUT,
    @cBrandName char(16)
    As
    SELECT @cBrandName = cBrandName
    FROM ToyBrand WHERE cBrandId = @cBrandId
    ```

    After successful creation of the procedure the following code was executed.

    ```
    DECLARE @cBrandName, char(16)
    Exec prcBrand '001', @cBrandName OUTPUT
    SELECT @cBrandName
    ```

    The code generates an error instead of displaying the required brand name. Detect and rectify the code.

15. The following code was used to create a procedure for adding the details of a new category to the category table:

    ```
    CREATE PROCEDURE prcAddCategory @CategoryId char(3),
    @Category char(20), @Descrip char(30)
    As
    INSERT Category VALUES ('CategoryId', 'Category', 'Descrip')
    ```

    After successful creation of the procedure the following code was used to execute the procedure for inserting a new category:

    ```
    Exec prcAddCategory '017', 'War Games', 'A wide range of toy guns'
    ```

    The above code generated an error and aborted. Detect and rectify the above code.