Database Programming

Exercise 1

Write a SP add_musical_style to add a new musical style, given the stylename of the new musical style (= parameter).

First check if the stylename doesn't already exist. If so throw an exception.

Use an output parameter to pass the result back to the user: 1 if the insert succeeded and 0 if the insert didn't succeed

Use the following testcode

```
-- Testcode 1: add Dance music => no problem
begin try
	begin transaction
	DECLARE @result tinyint
	EXEC add_musical_style 'Dance music', @result
	print 'Dance music inserted or not: ' + str(@result)
	select * from Musical_Styles
	rollback;
end try
begin catch
	DECLARE @e int;
	SET @e = ERROR_NUMBER();
	PRINT N'Error Procedure = ' + ERROR_PROCEDURE();
	PRINT N'Error Message = ' + ERROR_MESSAGE();

end catch

-- Testcode 2: add Classical => Error because Classical already exists

begin try
	begin transaction
	DECLARE @result tinyint
	EXEC add_musical_style 'Classical', @result
	print 'Classical inserted or not: ' + str(@result)
	select * from Musical_Styles
	rollback
end try
begin catch
	DECLARE @e int;
	SET @e = ERROR_NUMBER();
	PRINT N'Error Procedure = ' + ERROR_PROCEDURE();
	PRINT N'Error Message = ' + ERROR_MESSAGE();
end catch
```

Exercise 2

Write a SP delete_musical_style to delete a new musical style, given the stylename of the new musical style (= parameter).

First check if the stylename exists. If not so throw an error.

Then check if there are Entertainers with this musical style. If so, throw an error

Then check if there are Customers with this musical style. If so, throw an error

Delete the musical style

Use an output parameter to pass the result back to the user: 1 if the delete succeeded and 0 if the delete didn't succeed

Write testcode to try to delete the following styles: Chamber Music, Elvis, Variety

Exercise 3

Write a SP add_musical_preference to add a new musical preference given a customerid, a stylename and a preferenceSeq (= parameters)

Check if the customer exists. If not, throw an error

Check if the stylename exists. If not, throw an error

Check the preferenceSeq: if the customer already has a stylename with this preferenceSeq: this stylename and the subsequent styles get preferenceSeq + 1

Use an output parameter to pass the result back to the user: 1 if the insert succeeded and 0 if the insert didn't succeed

Write testcode for the following cases:

- for customer 10005 add Variety with PreferenceSeq = 3
- for customer 10007 add Variety with PreferenceSeq = 1

Exercise 4

Write a SP add_entertainer_style to add a new Entertainer_Style, given the entertainerid, the stylename and the stylestrength (= parameters)

Check if the given entertainer exists. If not, throw an error

Check if the given stylename exists. If not, throw an error

Check if the entertainer not already has this style. If so, throw an error

Check if the given stylestrength is a value between 1 and 3

If the given stylestrength is already taken for this entertainerid: the other stylestrengths are getting a new value = old value + 1. Make sure there are no more than 3 stylestrengths per entertainer

Use an output parameter to pass the result back to the user: 1 if the insert succeeded and 0 if the insert didn't succeed

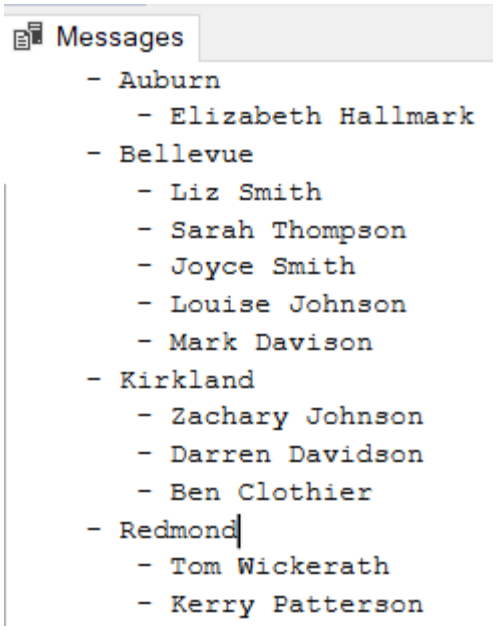
Write testcode for the following cases:

- Add entertainer with entertainerid = 1020 + stylename = chamber music + stylestrength = 1
- Add entertainer with entertainerid = 1003 + stylename = disco + stylestrength = 3
- Add entertainer with entertainerid = 1005 + stylename = Jazz + stylestrength = 1
- Add entertainer with entertainerid = 1005 + stylename = Jazz + stylestrength = 4
- Add entertainer with entertainerid = 1005 + stylename = Folk + stylestrength = 2
- Add entertainer with entertainerid = 1003 + stylename = Folk + stylestrength = 3

Exercise 5

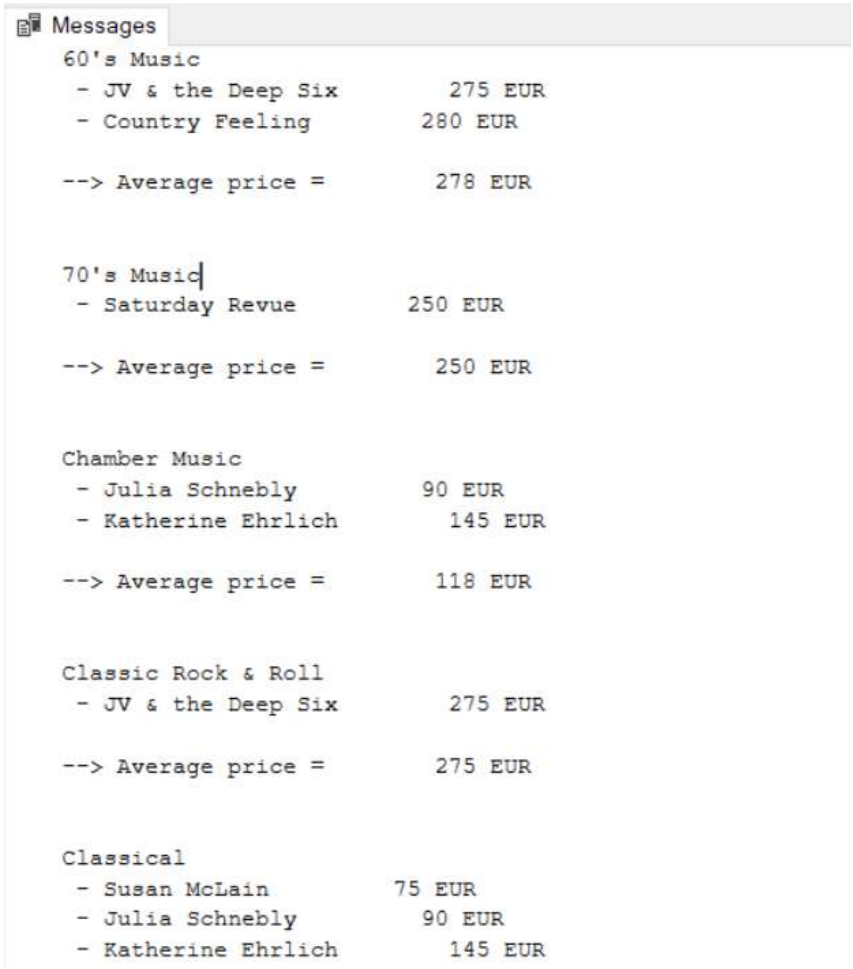
List all customers per city, sorted in alphabetical order.

The image below shows only a part of the result.



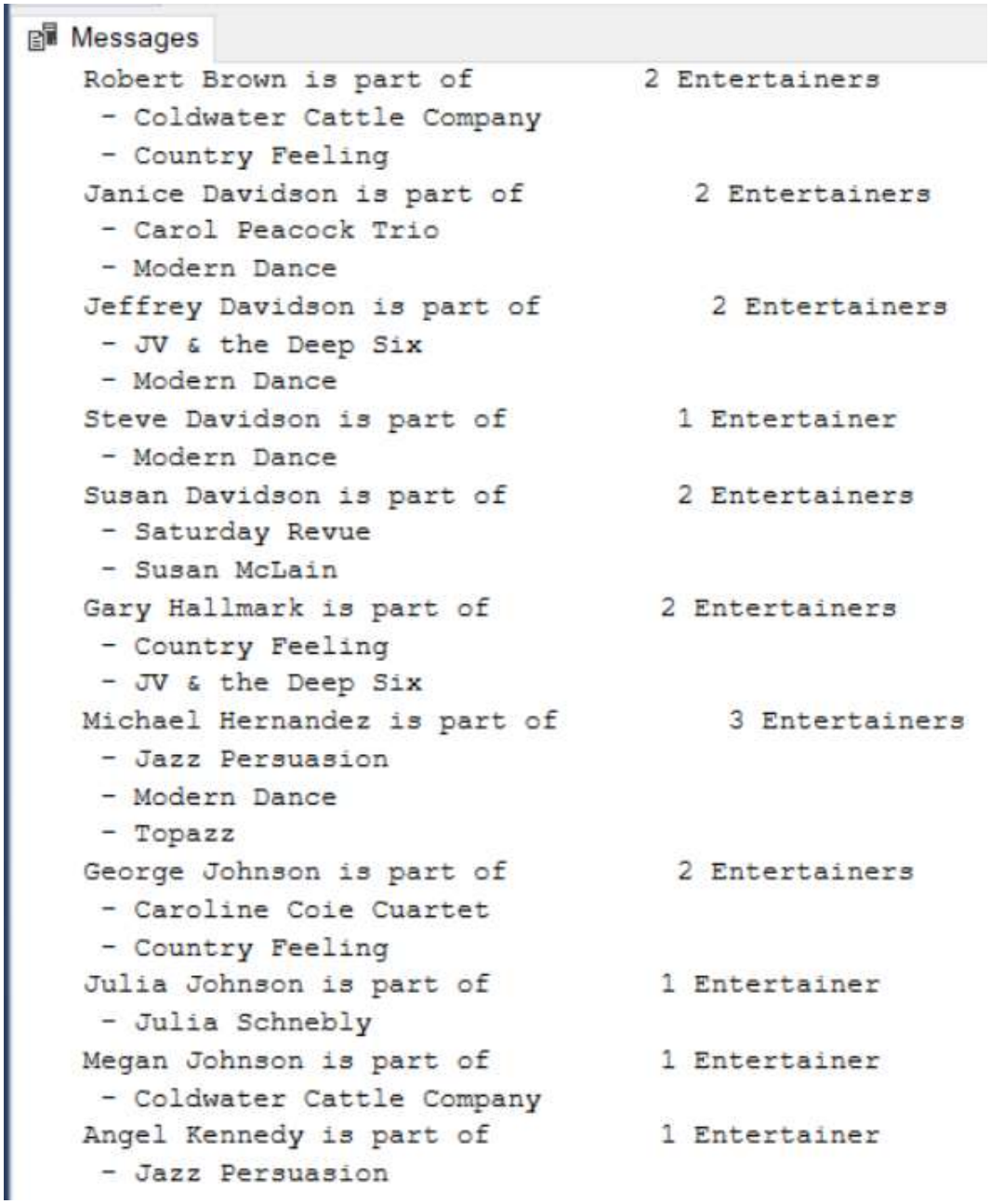
Exercise 6

List all artists per Entertainer_Styles, sorted from cheaper to more expensive.
If there is an Entertainer_Style for which there are no artists, it should not appear in the overview
Also calculate the average price at the end for each Entertainer_Style
The image below shows only a part of the result.



Exercise 7

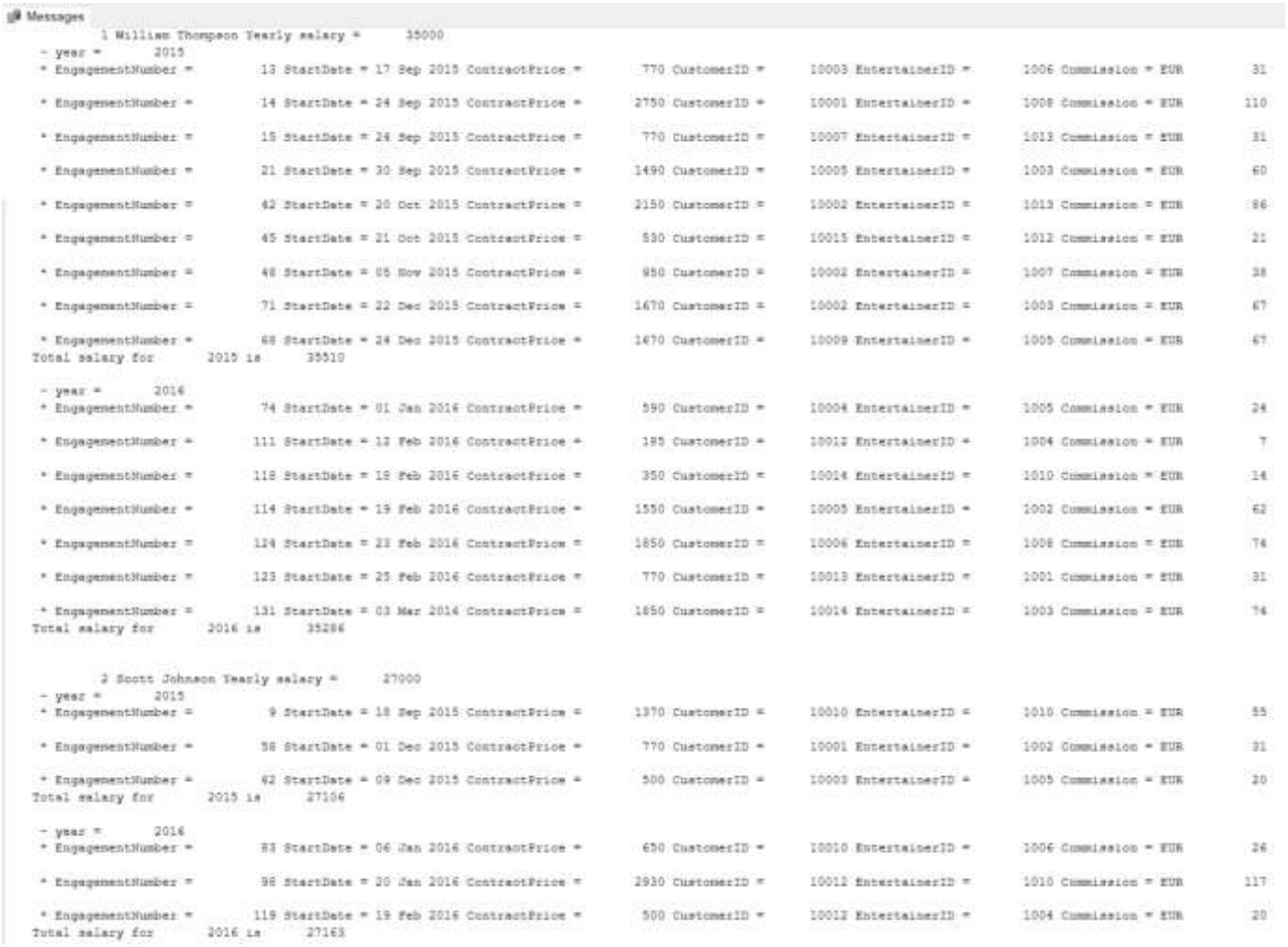
Give per Member the number of Entertainers a Member is part of and the list of the Entertainers of which a Member is a part of
Notice the difference between e.g. Robert Brown is part of 2 Entertainer**s** and Steve Davidson is part of 1 Entertainer
The image below shows only a part of the result.



Exercise 8

List the Engagements per Agent and what the Agent has earned per year
To add the the small sentence 'Total salary for 2015 is 35510' (for example), you have to add an IF statement after `FETCH NEXT FROM cursor_2 INTO …`

The image below shows only a part of the result.



Exercise 9

If there is an update of the Musical_Preferences table where the PreferenceSeq is adjusted, it must be checked whether the new PreferenceSeq is an allowed value (between 1 and 3)

If not, the transaction must be rolled back and an error should be thrown

In the testcode below, CustomerID = 10001 and StyleID = 10 get's PreferenceSeq = 10

```
begin try
	begin transaction

	UPDATE Musical_Preferences
	SET PreferenceSeq = 10
	WHERE CustomerID = 10001 and StyleID = 10

	print 'Update PreferenceSeq = 10 for CustomerID = 10001 and styleID = 10'

	select * from Musical_Preferences
	WHERE CustomerID = 10001

	rollback;
end try
begin catch
	DECLARE @e int;
	SET @e = ERROR_NUMBER();
	PRINT N'Error Procedure = ' + ERROR_PROCEDURE();
	PRINT N'Error Message = ' + ERROR_MESSAGE();
end catch
```

Exercise 10

If a record is removed from the Musical_Preferences table for a specific customer (for example CustomerID = 10007 + StyleID = 4 + PreferenceSeq = 2), then the PreferenceSeq (for example with value 3) of the subsequent records should be decreased by 1

Write testcode for the following examples

CustomerID	StyleID	PreferenceSeq		CustomerID	StyleID	PreferenceSeq
~~10007~~	~~4~~	~~2~~		10007	8	1
10007	8	1	⇨	10007	19	2
10007	19	3				

CustomerID	StyleID	PreferenceSeq		CustomerID	StyleID	PreferenceSeq
10007	4	2		10007	4	1
~~10007~~	~~8~~	~~1~~	⇨	10007	19	2
10007	19	3				

CustomerID	StyleID	PreferenceSeq		CustomerID	StyleID	PreferenceSeq
10007	4	2		10007	4	2
10007	8	1	⇨	10007	19	1
~~10007~~	~~19~~	~~3~~				

Exercise 11

If a record is inserted into the Musical_Preferences table for a specific customer (for example CustomerID = 10007 + StyleID = 5 + PreferenceSeq = 2), then the PreferenceSeq (for example with value 2 and 3) of the subsequent records should be increased by 1. All records with PreferenceSeq > 3 should be removed.

Is it useful to check whether the new combination of CustomerID and StyleID already exists in the table?

Write testcode for the following examples

CustomerID	StyleID	PreferenceSeq		CustomerID	StyleID	PreferenceSeq
10007	4	2		10007	4	3
10007	8	1	⇨	10007	8	1
10007	19	3		~~10007~~	~~19~~	~~3~~
10007	5	2		10007	5	2

CustomerID	StyleID	PreferenceSeq		CustomerID	StyleID	PreferenceSeq
10007	4	2		10007	4	2
10007	8	1	⇨	10007	8	1

10007	19	3	~~10007~~	~~19~~	~~3~~
10007	5	3	10007	5	3

CustomerID	StyleID	PreferenceSeq		CustomerID	StyleID	PreferenceSeq
10007	4	2		10007	4	3
10007	8	1	⇨	10007	8	2
10007	19	3		~~10007~~	~~19~~	~~3~~
10007	5	1		10007	5	1

Exercise 12

When a Member is added to an Entertainer, then it must be ensured that the EntPricePerDay is adjusted accordingly so that the price per member remains the same.

E.g.: Entertainer 1002 now has 2 members and the EntPricePerDay is 120 EUR. If an extra member is added, the EntPricePerDay should become 180 EUR.