KONSTANTINOS PAGANOPOULOS

01769789

LOGISTICS AND SUPPLY CHAIN ANALYTICS

INDIVIDUAL ASSIGNMENT

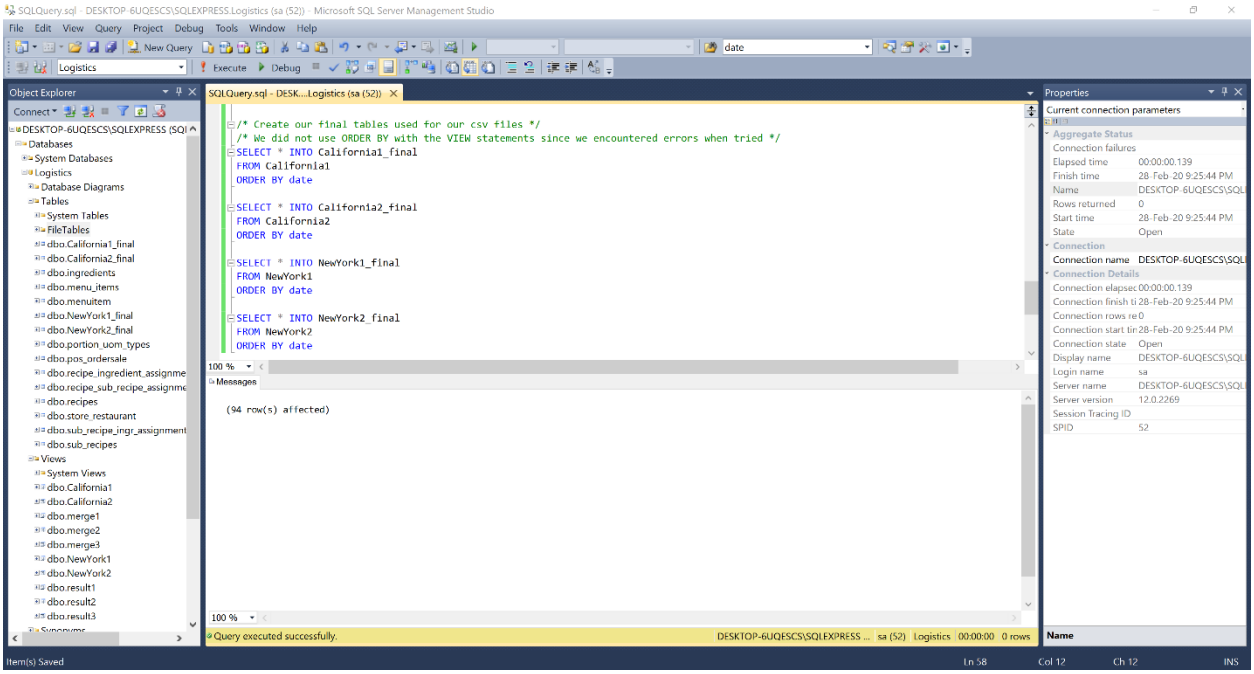
10/03/2020

INTRODUCTION

In this report solutions for individual assignment of "Logistics and Supply Chain Analytics MSc Business Analytics 2019/20" were provided. The data manipulation problem (correct merging of the tables) was solved using SQL and the Microsoft SQL Server Studio. The solutions include SQL code for solving a data wrangling problem, in which we have to merge separate tables according to their correspondence to the others.

DATA WRANGLING

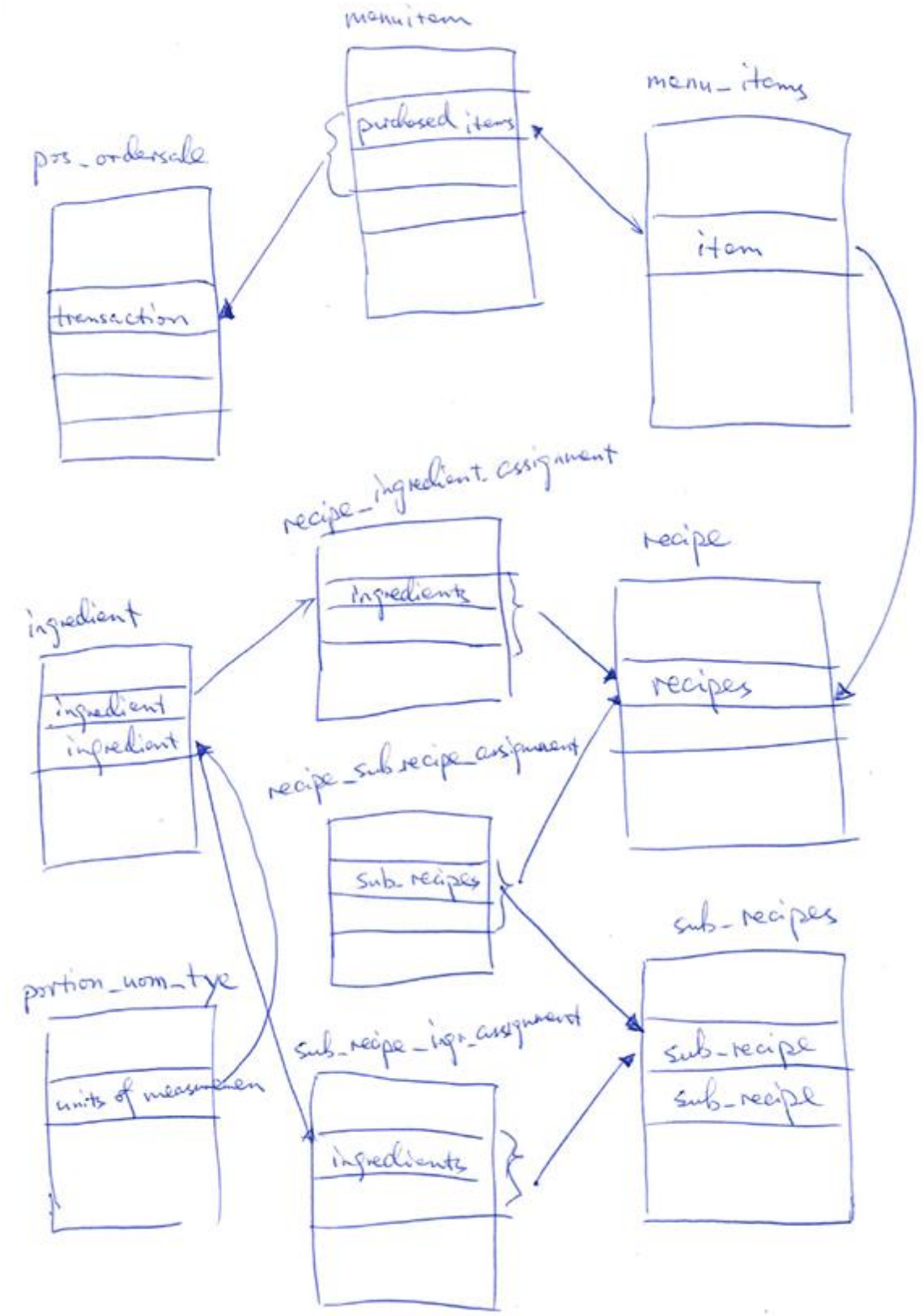
In order to make data more valuable to us and more specifically to store the total lettuce sales in 4 csv files (one for each store) according to date, we used the Microsoft SQL Server Studio.



Picture 1: Microsoft SQL Server Studio

We first created a database ("Logistics"), and all of the eleven tables (even if we eventually did not need all of them). After that, we imported the initial csv files to them and ran the following sql script.

Note that comments are provided within the script to explain parts of it that may not seem very easy for the reader to understand, as well as an ER diagram to explain how the merging of the tables was done.



Picture 2: ER Diagram

```sql
/* First total quantity sum */
CREATE VIEW merge1 AS(
SELECT
	portion_uom_types.PortionTypeDescription, ingredients.IngredientId,
sub_recipe_ingr_assignments.SubRecipeId,
	sub_recipe_ingr_assignments.Quantity, recipe_sub_recipe_assignments.RecipeId,
recipe_sub_recipe_assignments.Factor,
	(CAST(sub_recipe_ingr_assignments.Quantity AS numeric) *
CAST(recipe_sub_recipe_assignments.Factor AS numeric)) AS sum1
FROM ingredients
INNER JOIN sub_recipe_ingr_assignments
ON ingredients.IngredientId = sub_recipe_ingr_assignments.IngredientId
INNER JOIN recipe_sub_recipe_assignments
ON recipe_sub_recipe_assignments.SubRecipeId = sub_recipe_ingr_assignments.SubRecipeId
INNER JOIN portion_uom_types
ON ingredients.PortionUOMTypeId = portion_uom_types.PortionUOMTypeId
WHERE IngredientName LIKE 'Lettuce%'
)

CREATE VIEW result1 AS(
SELECT RecipeId, sum(sum1) AS lettuce_sum
FROM merge1
GROUP BY RecipeId
)

CREATE VIEW merge2 AS(
SELECT recipe_ingredient_assignments.RecipeId ,
sum(CAST(recipe_ingredient_assignments.Quantity AS numeric)) AS sum1
FROM ingredients
INNER JOIN recipe_ingredient_assignments
ON recipe_ingredient_assignments.IngredientId = ingredients.IngredientId
WHERE IngredientName LIKE 'Lettuce%'
GROUP BY recipe_ingredient_assignments.RecipeId
)

/* We use UNION to combine the result set of multiple SELECT statements. We use ALL to
allow duplicate values */
CREATE VIEW result2 AS(
SELECT * FROM result1
UNION ALL SELECT * FROM merge2
)

CREATE VIEW result3 AS(
SELECT RecipeId, sum(lettuce_sum) AS sum1
FROM result2
GROUP BY RecipeId
)

/* Second total quantity sum */
CREATE VIEW merge3 AS(
SELECT menuitem.*, menu_items.MenuItemId, result3.sum1 * menuitem.quantity AS lettuce_sum
FROM result3
INNER JOIN menu_items
ON result3.RecipeId = menu_items.RecipeId
INNER JOIN menuitem
ON menuitem.id = menu_items.MenuItemId
)
```

```
/* Create our final views used for our csv files */
/* We also did not use tables since we encountered errors */
CREATE VIEW California1 AS(
SELECT date, sum(lettuce_sum) AS lettuce
FROM merge3
WHERE StoreNumber = 46673
GROUP BY date, StoreNumber
)

CREATE VIEW California2 AS(
SELECT date, sum(lettuce_sum) AS lettuce
FROM merge3
WHERE StoreNumber = 4904
GROUP BY date, StoreNumber
)

CREATE VIEW NewYork1 AS(
SELECT date, sum(lettuce_sum) AS lettuce
FROM merge3
WHERE StoreNumber = 12631
GROUP BY date, StoreNumber
)

CREATE VIEW NewYork2 AS(
SELECT date, sum(lettuce_sum) AS lettuce
FROM merge3
WHERE StoreNumber = 20974
GROUP BY date, StoreNumber
)

/* Create our final tables used for our csv files */
/* We did not use ORDER BY with the VIEW statements since we encountered errors when
tried */
SELECT * INTO California1_final
FROM California1
ORDER BY date

SELECT * INTO California2_final
FROM California2
ORDER BY date

SELECT * INTO NewYork1_final
FROM NewYork1
ORDER BY date

SELECT * INTO NewYork2_final
FROM NewYork2
ORDER BY date
```

At the end, we exported our 4 final tables to our csv files.

CONCLUSION

In this report solutions for individual assignment of "Logistics and Supply Chain Analytics MSc Business Analytics 2019/20" were provided. The data manipulation problem (correct merging of the tables) was solved using SQL and the Microsoft SQL Server Studio.