“УКРАЇНСЬКИЙ КАТОЛИЦЬКИЙ УНІВЕРСИТЕТ”



**Звіт**

про виконання лабораторної роботи № 3

з курсу «Організація баз даних»

на тему «Написання запитів SELECT»

Варіант № 8

Виконав:

Студент групи ПКН15/Б

Карпінський Максим

Перевірив:

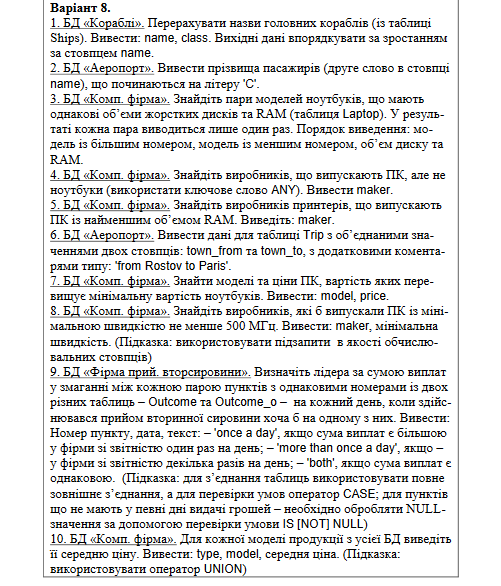
Павельчак А.Г.

Львів - 2017

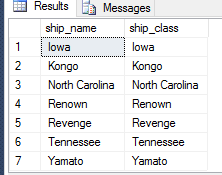
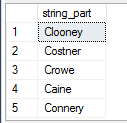
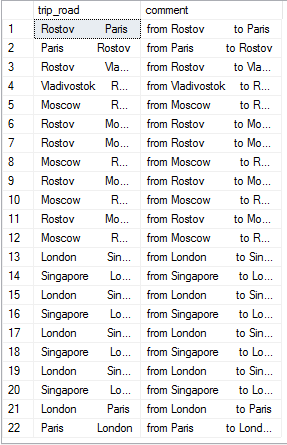
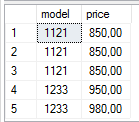
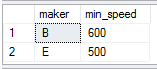
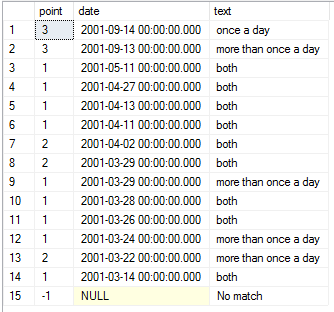
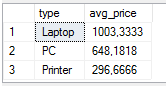
**Мета роботи:** написання SQL - запитів для вибірки необхідних даних і статистичної інформації з баз даних, що вказані в додатку, та виконання цих запитів до БД у графічному інструменті Management Studio, перевірити на працездатність та правильність отримуваних даних.

Практика написання оптимальних та працездатних SQL – запитів.

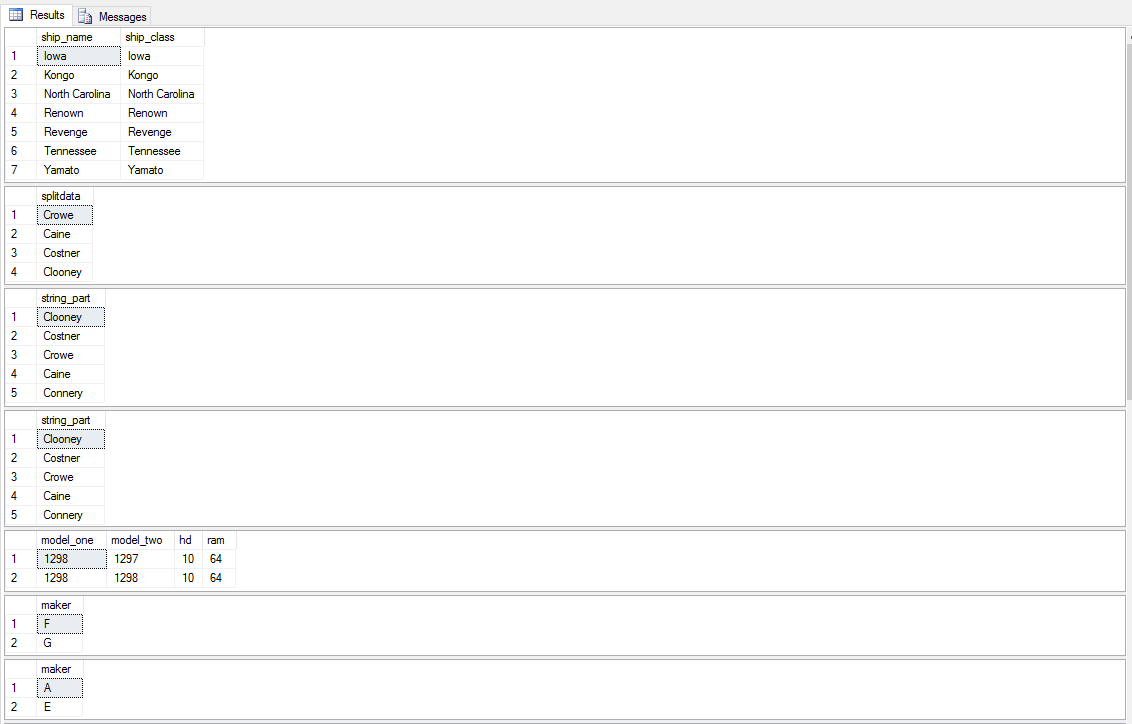
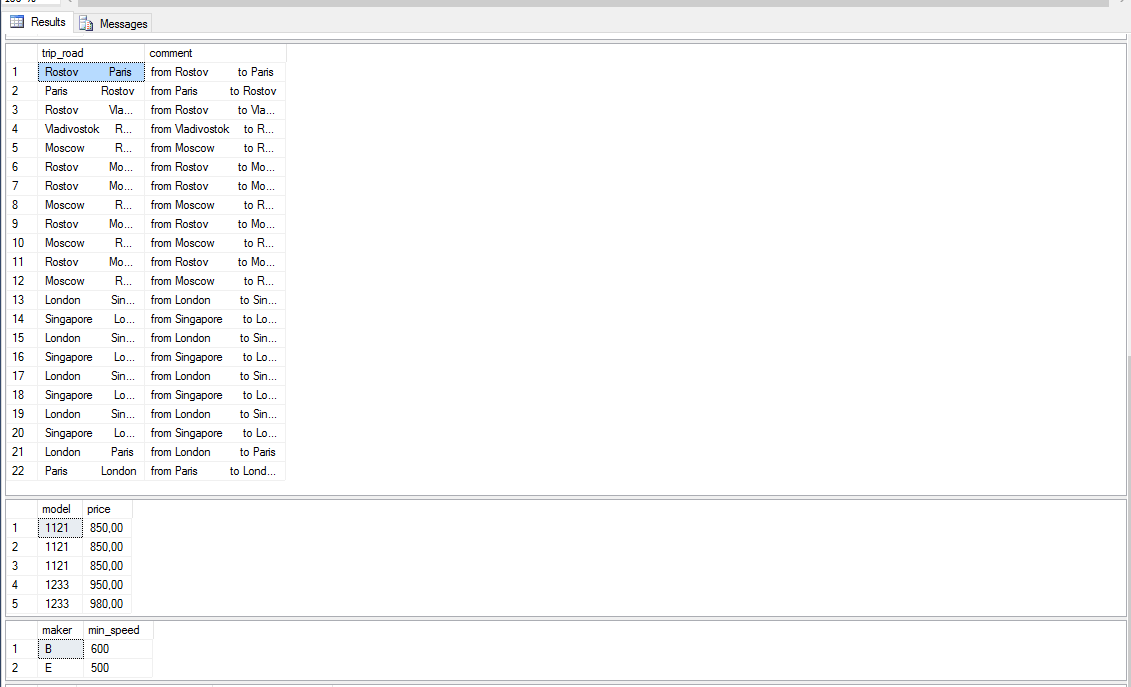
**Завдання до лабораторної роботи**

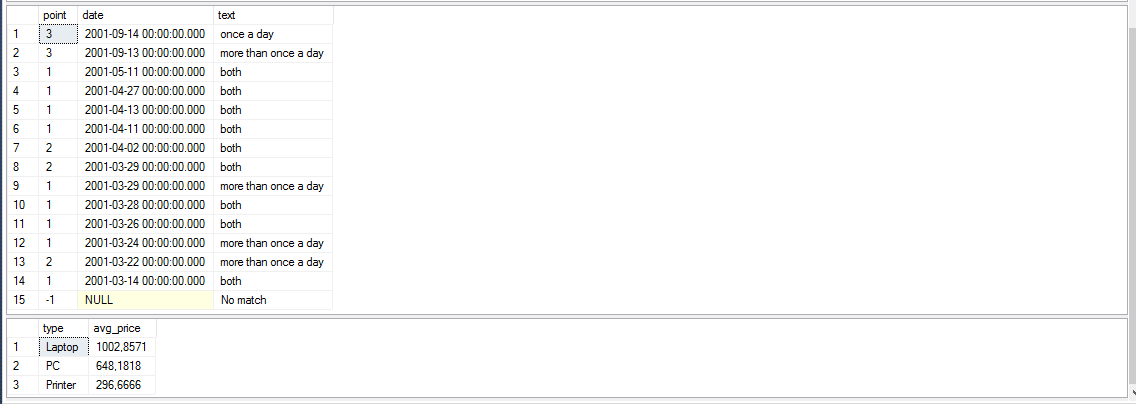


**Створені SQL – запити до БД та результати їхнього виконання**

1. USE Labor\_SQL;
2. GO
3. IF EXISTS(SELECT \* FROM sysobjects WHERE xtype = 'TF' AND name = 'SO\_SPLIT\_STRING')
4. BEGIN
5. DROP FUNCTION [dbo].[SO\_SPLIT\_STRING]
6. END
7. GO
8. CREATE FUNCTION [dbo].[SO\_SPLIT\_STRING]
9. (
10. @string NVARCHAR(MAX),
11. @delimiter CHAR(1)
12. )
13. RETURNS @output TABLE(splitdata NVARCHAR(MAX))
14. BEGIN
15. SET @string += @delimiter
16. DECLARE @start INT, @end INT
17. SELECT @start = 1, @end = CHARINDEX(@delimiter, @string)
18. WHILE @start < LEN(@string) + 1
19. BEGIN
20. IF @end = 0
21. SET @end = LEN(@string) + 1
22. INSERT INTO @output (splitdata)
23. VALUES (SUBSTRING(@string, @start, @end - @start))
24. SET @start = @end + 1
25. SET @end = CHARINDEX(@delimiter, @string, @start)
26. END
27. RETURN
28. END
29. GO
30. IF EXISTS(SELECT \* FROM sysobjects WHERE xtype = 'TF' AND name = 'SO\_GET\_STRING\_PART')
31. BEGIN
32. DROP FUNCTION [dbo].[SO\_GET\_STRING\_PART]
33. END
34. GO
35. CREATE FUNCTION [dbo].[SO\_GET\_STRING\_PART]
36. (
37. @string NVARCHAR(MAX),
38. @delimiter CHAR(1),
39. @part\_index INT
40. )
41. RETURNS @output TABLE(string\_part NVARCHAR(MAX))
42. BEGIN
43. SET @string += @delimiter
44. DECLARE @start INT, @end INT, @current\_index INT
45. SELECT @start = 1, @end = CHARINDEX(@delimiter, @string), @current\_index = 0
46. WHILE @start < LEN(@string) + 1 AND @current\_index != @part\_index
47. BEGIN
48. IF @end = 0
49. SET @end = LEN(@string) + 1
50. SET @start = @end + 1
51. SET @end = CHARINDEX(@delimiter, @string, @start)
52. SET @current\_index += 1
53. END
54. INSERT INTO @output (string\_part)
55. VALUES (SUBSTRING(@string, @start, @end - @start))
57. RETURN
58. END
59. GO
60. IF EXISTS(SELECT \* FROM sysobjects WHERE xtype = 'TF' AND name = 'SO\_CUT\_STRING\_TO')
61. BEGIN
62. DROP FUNCTION [dbo].[SO\_CUT\_STRING\_TO]
63. END
64. GO
65. CREATE FUNCTION [dbo].[SO\_CUT\_STRING\_TO]
66. (
67. @string NVARCHAR(MAX),
68. @delimiter CHAR(1),
69. @part\_index INT
70. )
71. RETURNS @output TABLE(string\_part NVARCHAR(MAX))
72. BEGIN
73. DECLARE @start INT, @end INT, @current\_index INT
74. SELECT @start = 1, @end = CHARINDEX(@delimiter, @string), @current\_index = 0
75. WHILE @start < LEN(@string) + 1 AND @current\_index != @part\_index
76. BEGIN
77. IF @end = 0
78. SET @end = LEN(@string) + 1
79. SET @start = @end + 1
80. SET @end = CHARINDEX(@delimiter, @string, @start)
81. SET @current\_index += 1
82. END
83. INSERT INTO @output (string\_part)
84. VALUES (SUBSTRING(@string, @start, LEN(@string)))
86. RETURN
87. END
88. GO
89. /\*
90. 1) БД «Кораблі».
91. Перерахувати назви головних кораблів (із таблиці Ships).
92. Вивести: name, class.
93. Вихідні дані впорядкувати за зростанням за стовпцем name.
94. \*/
95. SELECT [Ships].[name] AS ship\_name, [Ships].[class] AS ship\_class
96. FROM [Ships]
97. WHERE [Ships].name = [Ships].class
98. ORDER BY ship\_name;
99. /\*
100. 2) БД «Аеропорт».
101. Вивести прізвища пасажирів (друге слово в стовпці name), що починаються на літеру 'С'.
102. \*/
103. -- CASE 1 - Not giving the correct data because of middle or tripple names in name field. Look at CASE 2 for more correct data.
104. WITH SplitNameData AS
105. (
106. SELECT ROW\_NUMBER() OVER(ORDER BY [Passenger].[name] DESC) AS id, [passenger\_name\_split].splitdata
107. FROM [Passenger]
108. CROSS APPLY (SELECT \* FROM [SO\_SPLIT\_STRING]([Passenger].[name], ' ')) AS [passenger\_name\_split]
109. )
110. SELECT [splitdata]
111. FROM [SplitNameData]
112. WHERE [id] % 2 = 0 AND [splitdata] LIKE 'C%';
113. -- CASE 2 - Loses some last names due to midle or tripple name because gets 2 word in column, though data is correct, but not full. Look at CASE 3 for more correct data.
114. SELECT [passenger\_last\_name].[string\_part]
115. FROM [Passenger]
116. CROSS APPLY (SELECT \* FROM [SO\_GET\_STRING\_PART]([Passenger].[name], ' ', 1)) AS [passenger\_last\_name]
117. WHERE [passenger\_last\_name].[string\_part] LIKE 'C%'
118. GO
119. -- CASE 3 - Gives the most precise and correct data.
120. SELECT [passenger\_full\_last\_name].[string\_part]
121. FROM [Passenger]
122. CROSS APPLY (SELECT \* FROM [SO\_CUT\_STRING\_TO]([Passenger].[name], ' ', 1)) AS [passenger\_full\_last\_name]
123. WHERE [passenger\_full\_last\_name].[string\_part] LIKE 'C%'
124. GO
125. /\*
126. 3) БД «Комп. фірма».
127. Знайдіть пари моделей ноутбуків, що мають однакові об’єми жорстких дисків та RAM (таблиця Laptop).
128. У результаті кожна пара виводиться лише один раз.
129. Порядок виведення: модель із більшим номером, модель із меншим номером, об’єм диску та RAM.
130. \*/
131. SELECT DISTINCT ((CAST([LP\_one].[model] AS INT) + CAST([LP\_two].[model] AS INT)) + ABS(CAST([LP\_one].[model] AS INT) - CAST([LP\_two].[model] AS INT))) / 2 AS [model\_one],
132. ((CAST([LP\_one].[model] AS INT) + CAST([LP\_two].[model] AS INT)) - ABS(CAST([LP\_one].[model] AS INT) - CAST([LP\_two].[model] AS INT))) / 2 AS [model\_two],
133. [LP\_one].hd,
134. [LP\_one].ram
135. FROM [Laptop] AS LP\_one
136. JOIN [Laptop] AS LP\_two
137. ON [LP\_one].ram = [LP\_two].ram AND [LP\_one].hd = [LP\_two].hd AND [LP\_one].code != [LP\_two].code;
138. /\*
139. 4) БД «Комп. фірма».
140. Знайдіть виробників, що випускають ПК, але не ноутбуки (використати ключове слово ANY).
141. Виведіть: maker.
142. \*/
143. SELECT [PR].[maker]
144. FROM [Product] AS [PR]
145. WHERE NOT 'Laptop' = ANY(SELECT [Product].[type] FROM [Product] WHERE [Product].[maker] = [PR].[maker])
146. AND 'PC' = ANY(SELECT [Product].[type] FROM [Product] WHERE [Product].[maker] = [PR].[maker])
147. GROUP BY [PR].[maker];
148. /\*
149. 5) БД «Комп. фірма».
150. Знайдіть виробників принтерів, що випускають ПК із найменшим об’ємом RAM.
151. Виведіть: maker.
152. \*/
153. DECLARE @PCMinRam INT;
154. SELECT @PCMinRam = MIN([PC].ram) FROM [PC];
155. DECLARE @PrinterMakers TABLE (maker VARCHAR(1) UNIQUE);
156. INSERT INTO @PrinterMakers
157. SELECT [Product].[maker]
158. FROM [Product]
159. JOIN [Printer]
160. ON [Product].[model] = [Printer].[model]
161. GROUP BY [Product].[maker]
162. SELECT DISTINCT [Product].[maker]
163. FROM [Product]
164. JOIN [PC]
165. ON [Product].[model] = [PC].[model] AND [PC].[ram] = @PCMinRam AND [Product].[maker] IN (SELECT \* FROM @PrinterMakers);
166. /\*
167. 6) БД «Аеропорт».
168. Вивести дані для таблиці Trip з об’єднаними значеннями двох стовпців: town\_from та town\_to,
169. з додатковими коментарями типу: 'from Rostov to Paris'.
170. \*/
171. SELECT CONCAT([town\_from], ' ', [town\_to]) AS [trip\_road], CONCAT('from ', [town\_from], ' to ', [town\_to]) AS [comment]
172. FROM [Trip];
173. /\*
174. 7) БД «Комп. фірма».
175. Знайти моделі та ціни ПК, вартість яких перевищує мінімальну вартість ноутбуків.
176. Вивести: model, price.
177. \*/
178. DECLARE @LaptopMinValue INT;
179. SELECT @LaptopMinValue = MIN([price]) FROM [Laptop];
180. SELECT [model], [price]
181. FROM [PC]
182. WHERE [price] > @LaptopMinValue;
183. /\*
184. 8) БД «Комп. фірма».
185. Знайдіть виробників, які б випускали ПК із мінімальною швидкістю не менше 500 МГц.
186. Вивести: maker, мінімальна швидкість.
187. (Підказка: використовувати підзапити в якості обчислювальних стовпців)
188. \*/
189. SELECT [prt].[maker], MIN([pc].[speed]) AS min\_speed
190. FROM [PC] AS [pc]
191. JOIN [Product] AS [prt]
192. ON [pc].[model] = [prt].[model]
193. GROUP BY [maker]
194. HAVING MIN([pc].[speed]) >= 500;
195. /\*
196. 9) БД «Фірма прий. вторсировини».
197. Визначіть лідера за сумою виплат у змаганні між кожною парою пунктів
198. з однаковими номерами із двох різних таблиць – Outcome та Outcome\_o – на кожний день,
199. коли здійснювався прийом вторинної сировини хоча б на одному з них.
200. Вивести: Номер пункту, дата, текст:
201. – 'once a day', якщо сума виплат є більшою у фірми зі звітністю один раз на день;
202. – 'more than once a day', якщо – у фірми зі звітністю декілька разів на день;
203. – 'both', якщо сума виплат є однаковою.
204. (Підказка: для з’єднання таблиць використовувати повне зовнішнє з’єднання,
205. а для перевірки умов оператор CASE;
206. для пунктів, що не мають у певні дні видачі грошей – необхідно обробляти NULL -
207. значення за допомогою перевірки умови IS [NOT] NULL)
208. \*/
209. SELECT
210. CASE
211. WHEN [any\_outcome].[point] IS NULL OR [day\_outcome].[point] IS NULL
212. THEN -1
213. ELSE [any\_outcome].[point]
214. END AS [point],
215. [any\_outcome].[date] AS [date],
216. CASE
217. WHEN [day\_outcome].[out] > [any\_outcome].[out]
218. THEN 'once a day'
219. WHEN [day\_outcome].[out] < [any\_outcome].[out]
220. THEN 'more than once a day'
221. WHEN [day\_outcome].[out] IS NULL OR [any\_outcome].[out] IS NULL
222. THEN 'No match'
223. ELSE 'both'
224. END AS [text]
225. FROM [Outcome\_o] AS [day\_outcome]
226. FULL OUTER JOIN (SELECT [point], [date], SUM([out]) AS [out] FROM [Outcome] GROUP BY [point], [date]) AS [any\_outcome]
227. ON [any\_outcome].[date] = [day\_outcome].[date] AND [any\_outcome].[point] = [day\_outcome].[point]
228. ORDER BY [date] DESC;
229. /\*
230. 10) БД «Комп. фірма».
231. Для кожної моделі продукції з усієї БД виведіть її середню ціну.
232. Вивести: type, model, середня ціна.
233. (Підказка: використовувати оператор UNION)
234. \*/
235. SELECT 'PC' AS [type], AVG([price]) AS [avg\_price]
236. FROM [PC]
237. UNION
238. SELECT 'Laptop' AS [type], AVG([price]) AS [avg\_price]
239. FROM [Laptop]
240. UNION
241. SELECT 'Printer' AS [type], AVG([price]) AS [avg\_price]
242. FROM [Printer];

**Результати виконання SQL-запитів**





**Висновок:** на цій лабораторній роботі я ознайомився із написанням SQL – запитів та їхнього оптимального використання до БД у графічному інтерфейсі Management Studio.