Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Інститут прикладного системного аналізу
Кафедра математичних методів системного аналізу

ЗВІТ

про виконання комп'ютерного практикуму №3
з дисципліни «Системи баз даних»

Виконав:

Студент III курсу
Групи КА-96
Володимир Шапошніков

Перевірила:

Афанасьєва І. В.

Київ – 2021

План виконання роботи

Відповідно до обраної теми курсового проекту необхідно попрацювати над створенням представлень:

- Робота із однією таблицею;
- Робота з кількома таблицями;
- Робота із числами;
- Робота із датами.

Хід виконання роботи

Робота із однією таблицею.

№1. Представлення, яке повертає всі дані таблиці.

```
CREATE VIEW view1
AS SELECT * FROM items;
SELECT * FROM view1;
```

item_id	amount	customer_id
44	108	2
81	0	1
193	47	3
377	829	1
1839	0	3

№2. Представлення, яке повертає рядки, що задовольняють наперед заданій умові.

```
CREATE VIEW view2
AS SELECT * FROM items c WHERE c.amount > 100;
SELECT * FROM view2;
```

item_id	amount	customer_id
44	108	2
377	829	1

№3. Представлення, яке повертає рядки, що задовольняють наперед заданим умовам.

```
CREATE VIEW view3
AS SELECT * from items t
WHERE t.amount > 100 AND t.amount < 250;
SELECT * FROM view3;
```

item_id	amount	customer_id
44	108	2

№4.Представлення, яке повертає конкретні стовпці.

```
CREATE VIEW view4
AS SELECT t.amount, t.customer_id FROM items t;
SELECT * FROM view4;
```

amount	customer_id
108	2
0	1
47	3
829	1
0	3

№5. Представлення, яке повертає конкретні стовпці, які мають зрозумілі та зручні для читання імена (псевдоніми).

```
CREATE VIEW view5
AS SELECT t.amount AS 'Amount:', t.customer_id AS 'Customer_ID:' FROM items t;
SELECT * FROM view5;
```

Amount:	Customer_ID:
108	2
0	1
47	3
829	1
0	3

№6. Представлення, яке використовує конкатенацію стовпців.

```
CREATE VIEW view6
AS SELECT t.amount, CONCAT(t.item_id,' ',t.amount) AS 'Item id and amount:' FROM
items t;
SELECT * FROM view6;
```

amount	Item id and amount:
108	44 108
0	81 0
47	193 47
829	377 829
0	1839 0

№7. Представлення, яке використовує вираз CASE.

```
CREATE VIEW view7
AS SELECT t.item_id, t.amount, t.customer_id, CASE
WHEN t.amount BETWEEN 0 AND 50 THEN 'Small turnover'
WHEN t.amount BETWEEN 50 AND 150 THEN 'Middle turnover'
WHEN t.amount BETWEEN 150 AND 2500 THEN 'High turnover'
END AS 'Amount range:' FROM items t;
SELECT * FROM view7;
```

item_id	amount	customer_id	Amount range:
44	108	2	Middle turnover
81	0	1	Small turnover
193	47	3	Small turnover
377	829	1	High turnover
1839	0	3	Small turnover

№8. Представлення, яке повертає обмежену кількість рядків.

```
CREATE VIEW view8
AS SELECT t.amount, t.item_id, t.customer_id FROM items t LIMIT 3;
SELECT * FROM view8;
```

amount	item_id	customer_id
108	44	2
0	81	1
47	193	3

№9.Представлення, яке повертає n випадкових рядків таблиці.

```
CREATE VIEW view9
AS SELECT t.item_id, CONCAT(t.amount,' ',t.item_id) data
FROM items t ORDER BY RAND() LIMIT 6;
SELECT * FROM view9;
```

item_id	data
81	0 81
44	108 44
193	47 193
377	829 377
1839	0 1839

№10. Представлення з пошуком значень NULL.

```
CREATE VIEW view10
AS SELECT t.item_id, t.amount, t.customer_id FROM items t WHERE t.amount IS NULL;
SELECT * FROM view10;
```

item_id	amount	customer_id

№11. Представлення з пошуком за шаблоном.

```
CREATE VIEW view11
AS SELECT * FROM items t WHERE t.amount NOT LIKE '800%';
SELECT * FROM view11
```

item_id	amount	customer_id
44	108	2
81	0	1
193	47	3
377	829	1
1839	0	3

№12. Представлення, в якому рядки відсортовано за деяким полем.

```
CREATE VIEW view12
AS SELECT * FROM items t ORDER BY t.item_id;
SELECT * FROM view12;
```

item_id	amount	customer_id
44	108	2
81	0	1
193	47	3
377	829	1
1839	0	3

№13. Представлення, в якому рядки відсортовано за більш ніж одним полем.

```
CREATE VIEW view13
AS SELECT * FROM items t ORDER BY t.item_id , t.amount;
SELECT * FROM view13;
```

item_id	amount	customer_id
44	108	2
81	0	1
193	47	3
377	829	1
1839	0	3

№14. Представлення, в якому рядки відсортовано за під-рядком (функція substring)

```
CREATE VIEW view14
AS SELECT * FROM items t ORDER BY substring(t.amount, 2, 3);
SELECT * FROM view14;
```

item_id	amount	customer_id
81	0	1
1839	0	3
44	108	2
377	829	1
193	47	3

№15. Представлення, в якому обробляються NULL значення при сортуванні.

```
CREATE VIEW view15
AS SELECT t.item_id, t.customer_id, coalesce(t.amount,'****')
FROM items t ORDER BY COALESCE(t.amount,'****');
SELECT * FROM view15;
```

item_id	customer_id	coalesce(t.amount,'****')
81	1	0
1839	3	0
44	2	108
193	3	47
377	1	829

№16. Представлення, в якому рядки відсортовано за залежністю даних від ключа.

```
CREATE VIEW view16
AS SELECT t.item_id, t.amount,
CASE
WHEN t.amount BETWEEN '0' AND '50'
THEN 'very low turnover' ELSE 'normal turnover'
END AS 'Amount order:' FROM items t;
SELECT * FROM view16;
```

item_id	amount	Amount order:
44	108	normal turnover
81	0	very low turnover
193	47	very low turnover
377	829	normal turnover
1839	0	very low turnover

Робота з кількома таблицями.

№17. Представлення, яке розміщує один набір рядків під іншим.

```
CREATE VIEW view17
AS SELECT t.item_id, t.amount
FROM items t UNION ALL
SELECT c.company_id, c.turnover FROM companies c;
SELECT * FROM view17;
```

item_id	amount
44	108
81	0
193	47
377	829
1839	0
1	2800
2	1300
3	3100
4	450
5	2150
6	900

№18. Представлення, яке поєднує пов’язані рядки.

```
CREATE VIEW view18
AS SELECT customers.lastname, companies.title FROM customers
INNER JOIN companies ON customers.company = companies.title;
SELECT * FROM view18;
```

lastname	title
Lovien	BestTech
Nozin	RoomBoom
Mozila	CPRComp
Gerber	LTDI
Bort	BestTech
Geran	Moch-KIEV
Shevich	Wow-Lviv

№19. Представлення, яке знаходить однакові рядки в двох таблицях.

```
CREATE VIEW view19v1
AS SELECT firstname, company
FROM user;
CREATE VIEW view19v2
AS SELECT c.firstname, c.company FROM customers c, view19v1 h
WHERE c.firstname = h.firstname
AND c.company = h.company;
SELECT * FROM view19v2;
```

firstname	company
Alex	BestTech
Nikita	RoomBoom
Vasya	CPRComp
Roma	LTDI
Dima	BestTech
Vitalii	Moch-KIEV
Danyl	Wow-Lviv

№20. Представлення, яке повертає записи із значеннями, яких немає в іншій таблиці.

```
CREATE VIEW view20
AS SELECT c.title, c.turnover FROM companies c
WHERE c.title NOT IN (SELECT a.title FROM achievements a);
SELECT * FROM view20
```

title	turnover
BestTech	2800

№21. Представлення, яке повертає записи із значеннями, для яких немає відповідності в іншій таблиці.

```
CREATE VIEW view21
AS SELECT c.*, p.title FROM customers c
RIGHT OUTER JOIN companies p ON (c.company=p.title);
SELECT * FROM view21;
```

lastname	firstname	id	company	phone	title	position
NULL	*NULL*	*NULL*	*NULL*	*NULL*	BestTech	*NULL*
NULL	*NULL*	*NULL*	*NULL*	*NULL*	RoomBoom	*NULL*

№22. Представлення, в якому реалізовано незалежне додавання об’єднань у запит.

```
CREATE VIEW view22 AS
SELECT t.item_id AS 'Item ID:', t.amount AS 'Amount:', c.id as 'Company ID:' ,
a.ach_id AS 'Achievement:'
FROM items t
JOIN achievements a ON (a.title=c.title)
ORDER BY t.item_id;
SELECT * FROM view22;
```

ID item:	Amount:	ID Company	Company
44	108	1	BestTech
81	0	4	LTDI
377	829	5	Moch-KIEV
1839	0	6	Wow-Lviv
193	47	2	RoomBoom

№23. Представлення, в якому реалізовано об’єднання з використанням агрегованих функцій.

```
CREATE VIEW view23 AS
SELECT COUNT(DISTINCT c.lastName) AS 'Amount of customers:', p.title AS 'Company:'
FROM customers c
JOIN companies p ON (c.company = p.title)
GROUP BY p.title;
SELECT * FROM view23;
```

Amount of customers:	Company:
2	BestTech
1	CPRComp
1	LTDI
1	Moch-KIEV
1	RoomBoom
1	Wow-Lviv

№24. Представлення, в якому реалізовано зовнішнє об’єднання з використанням агрегованих функцій.

```
CREATE VIEW view24 AS
SELECT COUNT(DISTINCT c.lastname) AS 'Amount of customers:', COALESCE(p.title,
'without company') AS 'Company:' FROM customers c LEFT JOIN companies p ON
(c.company = p.title)
GROUP BY p.title;
SELECT * FROM view24
```

Amount of customers:	Company:
2	BestTech
1	CPRComp
1	LTDI
1	Moch-KIEV
1	RoomBoom
1	Wow-Lviv

№25. Представлення, в якому відсутні дані в кількох таблицях.

```
CREATE VIEW view25 AS
SELECT c.lastname AS lastName, c.phone AS 'phone', c.firstname AS firstName,
p.title AS title, a.title AS achName FROM customers c
RIGHT JOIN companies p ON c.company=p.title
LEFT JOIN achievements a ON a.title=p.title;
SELECT * FROM view25;
```

lastName	phone	firstName	title	achName
Lovien	+380973839922	Alex	BestTech	*NULL*
Nozin	+380993539511	Nikita	RoomBoom	*NULL*
Mozila	+380990748183	Vasya	CPRComp	*NULL*
Gerber	+380678397744	Roma	LTDI	*NULL*
Bort	+380966883391	Dima	BestTech	*NULL*
Geran	+380958297485	Vitalii	Moch-KIEV	*NULL*
Shevich	+380603898503	Danyl	Wow-Lviv	*NULL*

Робота з масивами символів – рядками

№27. Представлення з проходом рядка.

```
CREATE VIEW view27 AS
SELECT substr(c.lastname, iter.pos -1, 1) AS 'Result:' FROM (SELECT
lastname FROM customers) c, (SELECT title AS ctit FROM companies) iter
WHERE iter.pos <=length(c.lastname) + 1;
SELECT * FROM view27;
```

Result:
B
o
p
M
k
m
B
B
t
B
V
i
B
N
e

№28. Представлення, в якому виводяться одиночні лапки.

```
CREATE VIEW view28
AS SELECT ' '' ' as 'Result:';
SELECT * FROM view28;
```

Result:
'

№29. Представлення, в якому видаляються всі непотрібні символи.

```
CREATE VIEW view29 AS
SELECT phone, REPLACE(phone,'+38','') AS 'without code' FROM customers;
SELECT * FROM view29;
```

phone	without code
+380973839922	0973839922
+380993539511	0993539511
+380990748183	0990748183
+380678397744	0678397744
+380966883391	0966883391
+380958297485	0958297485
+380603898503	0603898503

№30. Представлення, в якому розділяються числові та символьні дані

```
CREATE VIEW view30 AS
SELECT title, REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(studioNam e, 'b',
''), 'a',
''), 'l', ''), 'k', ''), 'o', ''), 'n', '') AS `numbers`,
REPLACE(REPLACE(studioName, '2', ''), '2',
'') AS `characters` FROM companies;
```

title	numbers	characters
Comp 32	32	Comp

№31.Представлення, в якому вибираються ініціали з імені.

```
CREATE VIEW view31 AS
SELECT 'Nikita Vovkin' as fullname, REPLACE(
REPLACE( REPLACE( REPLACE( REPLACE( REPLACE(
REPLACE(
REPLACE(
REPLACE('Nikita Vovkin', 'l', ''),
'a', ''), 'd', ''), 'e', ''), 't', ''), 'r', ''), 'o', ''), 'v', ''), '
','') AS `initials`
```

fullName	initials
Nikita Vovkin	N V

№32. Представлення, в якому перетворити рядок у список оператора IN із множиною значень.

```
CREATE VIEW view32
AS SELECT c.lastname, c.firstname, c.company FROM customers c
WHERE c.company IN(SELECT
substring_index(substring_index(val.str,',',iter.pos),',',-1) AS id FROM (SELECT
p.company_id pos FROM companies p) iter JOIN (SELECT '1, 2' str) val WHERE
iter.pos<=length(val.str)-length(replace(val.str,',',''))+1);
SELECT * FROM view32;
```

lastname	firstname	company
Lovien	Alex	BestTech
Nozin	Nikita	RoomBoom
Mozila	Vasya	CPRComp
Gerber	Roma	LTDI
Bort	Dima	BestTech
Geran	Vitalii	Moch-KIEV
Shevich	Danyl	Wow-Lviv

Робота з числами

№33. Представлення, в якому вивести мінімальне/максимальне значення у стовпці.

```
CREATE VIEW view33 AS
SELECT MIN(t.amount)AS 'Min amount:', MAX(t.amount) AS 'Max
amount:' FROM items t;
SELECT * FROM view33;
```

Параметры

Min amount:	Max amount:
0	829

№34. Представлення, яке повертає кількість записів у таблиці.

```
CREATE VIEW view34 AS
SELECT COUNT(*) AS 'Amount of customers:' FROM user; SELECT * FROM view34;
```

Amount of customers:
7

№35. Представлення, яке повертає кількість визначених (не-NULL) значень.

```
CREATE VIEW view35 AS
SELECT COUNT(title) AS 'Quantity of (not Null) company name (title):' FROM
companies;
SELECT * FROM view35;
```

Quantity of (not Null) company name (title):
6

№36. Представлення, в якому обчислити поточну суму

```
CREATE VIEW view35V1 AS
SELECT t.amount FROM items t;
SELECT SUM(amount) AS 'All amount' FROM items;
```

All amount
984

№37. Представлення, в якому обчислити кількість днів між двома датами.

```
CREATE VIEW view37v2 AS
SELECT a.title, DATEDIFF(a.g_date,a.e_date) AS 'time period' FROM achievements a;
SELECT * FROM view37v2;
```

title	time period
BestTech 1000 turnover ach!	82

№38. Представлення, в якому обчислити кількість повторів днів тижня протягом року

```
CREATE VIEW view38 AS
SELECT date_format( date_add( CAST( CONCAT(year(CURRENT_DATE),'- 01- 01') AS date),
interval t500.id-1 day), '%W') day, count(*) FROM t500
```

```
WHERE t500.id <= DATEDIFF( CAST( CONCAT(year(CURRENT_DATE)+1,'- 01-01') AS date),
CAST( CONCAT(year(CURRENT_DATE),'-01-01') AS date))
GROUP BY date_format( date_add( CAST( CONCAT(year(CURRENT_DATE),'-01-01') AS date),
interval t500.id-1 day), '%W');
SELECT * FROM view38
```

day	count(*)
Monday	13
Tuesday	7
Wednesday	24
Thursday	33
Friday	9
Saturday	18
Sunday	11

№39. Представлення, в якому визначити, чи рік високосний.

```
CREATE VIEW view39 AS
SELECT CASE WHEN (YEAR(CURRENT_DATE) % 4 = 0) AND ((YEAR(CURRENT_DATE) % 100 != 0)
OR (YEAR
(CURRENT_DATE) % 400 = 0)) THEN 'Pre-Previous year' ELSE 'No Pre-Previous year' END
AS '2020:';
SELECT * FROM view39
```

2020:
No Previous year

№40. Представлення, в якому визначити перший та останній день місяця.

```
CREATE VIEW view40 AS
SELECT DATE_add(current_date, interval -day(current_date)+1 day) AS '1-st day:',
DATE_add(date_add(current_date,interval 1 month),interval -day(current_date) day)
AS '2-nd day:';
SELECT * FROM view40;
```

1-st day:	2-nd day:
2022-08-01	2022-08-31

№41. Представлення, в якому створити календар.

```
CREATE VIEW view40 AS SELECT
min(case dw when 2 then dm end) as Monday, max(case dw when 3 then dm end) as
Tuesday, max(case dw when 4 then dm end) as Wednesday, min(case dw when 5 then dm
end) as Thursday, max(case dw when 6 then dm end) as Friday, max(case dw when 7
then dm end) as Saturday, max(case dw when 1 then dm end) as Sunday FROM (
select date_add(x.dy,interval t500.id-1 day) dy, x.mth
```

```
FROM(select date_add(current_date,interval -day(current_date)+1 day) as dy,
date_format(date_add(current_date,interval -
day(current_date)+1 day),'%m') as mth) x, t500 where t500.id<=31 and
date_format(date_add(x.dy,interval t500.id- 1 day),' %m')=x.mth )y )z
GROUP BY wk ORDER BY wk;
SELECT * FROM view40
```

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	NULL	NULL	NULL	NULL

№42. Представлення, в якому доповнити дати, що відсутні.

```
CREATE VIEW view42 AS
select * from (select adddate('1950-01-01',t4*10000 + t3*1000 + t2*100 + t1*10 +
t0)
union select 6 union select 7 union select 8 union select 9) t0, (select 0 t1 union
select 1 union select 2 union select 3 union select 4 union select 5 union select 6
union select 7 union select 8 union select 9) t1, (select 0 t2 union select 1 union
select 2 union select 3
union select 6 union select 7 union select 8 union select 9) t2, (select 0 t3 union
select 1 union select 2 union select 3 union select 4 union select 5 union select 6
union select 7 union select 8 union select 9) t3, (select 0 t4 union select 1 union
select 2 union select 3 union select 4 union select 5 union select 6 union select 7
union select 8 union select 9) t4) v
where selected_date between '2020-03-12' and '2020-04-01'; SELECT * FROM view42;
```

selected_date
2022-08-16
2022-08-06
2022-08-09
2022-08-03
2022-09-02
2022-08-23
2022-09-11
2022-09-05
2022-08-01
2022-08-09
2022-08-30
2022-09-12
2022-08-11

№42.Представлення, в якому виявити накладення діапазонів дат.

```
CREATE or REPLACE VIEW task43 AS
SELECT p.title, concat('company',m.company_id, ' re-comp ',a.company_id) AS reply
FROM companies p, companies m WHERE a.title = b.title
AND b.date_start >= a.date_start AND b.date_start <= a.date_end AND a.company_id !=
b.company_id;
SELECT * FROM task43;
```

company	re-comp

Висновки

У процесі виконання роботи я ознайомився з базовими та важливими функціями та можливостями MySQL, а також проводити маніпуляції з однією та більше таблицями. Також проводити роботу та маніпуляції з числовими та малими даними, виводити дати та обчислювати їх та проводити об'єднання таблиць за допомогою join і тд. У ході виконання роботи було створено 5 таблиці та виконано всі пункти.