**Методичні рекомендації**

**до виконання лабораторної роботи №№ 12-13**

Складні запити. Створення підзапитів.

Мета: Отримати навички створення підзапитів похідних таблиць, підзапитів з одним значенням, підзапитів в логічних виразах.

!!! Увага!!!

При виконанні лабораторної роботи необхідно реалізувати 4 запити:

1 Підзапити похідних таблиць

2 Підзапити з одним значенням

3 Підзапити в логічних вираженнях

4 Використання агрегатних функцій

ТЕОРЕТИЧНІ ВІДОМОСТІ ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ

1 Створення підзапитів

Підзапит - це запит усередині іншого запиту, тобто запит, у якому використаються результати іншого запиту. Іноді підзапити називають вкладеними запитами. Підзапити є новою можливістю в MySQL 4.1. Цієї можливості користувачі чекали давно. Підзапити не додають нових функціональних можливостей, але з ними запити часто виявляються більше зручними для читання, чим зі складними наборами умов об'єднання.

В MySQL були додані два основних види підзапитів:

* підзапити похідних таблиць;
* підзапити-вирази.

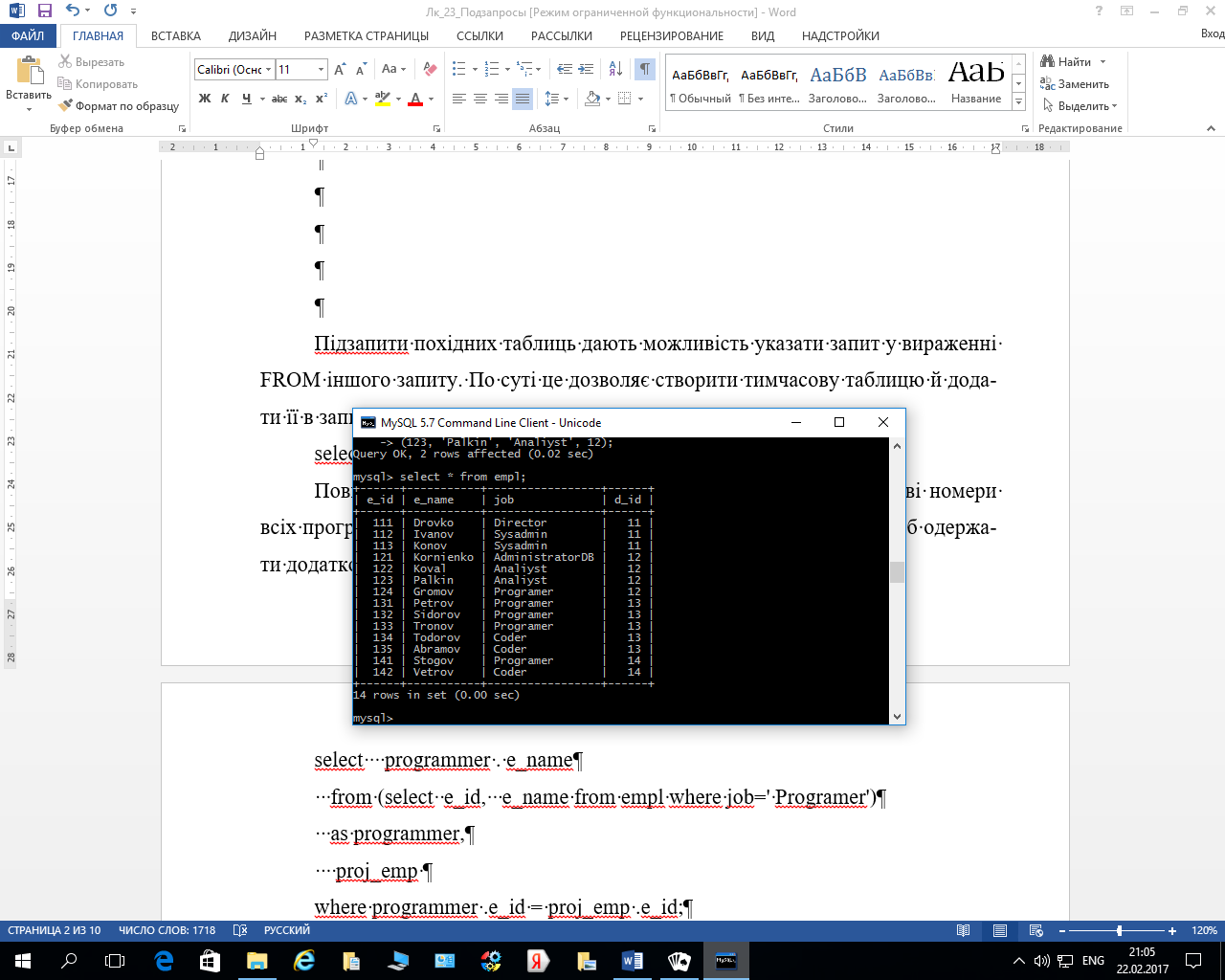
Підзапити-вирази застосовуються у вираженні WHERE оператора SELECT, Вони бувають двох типів:

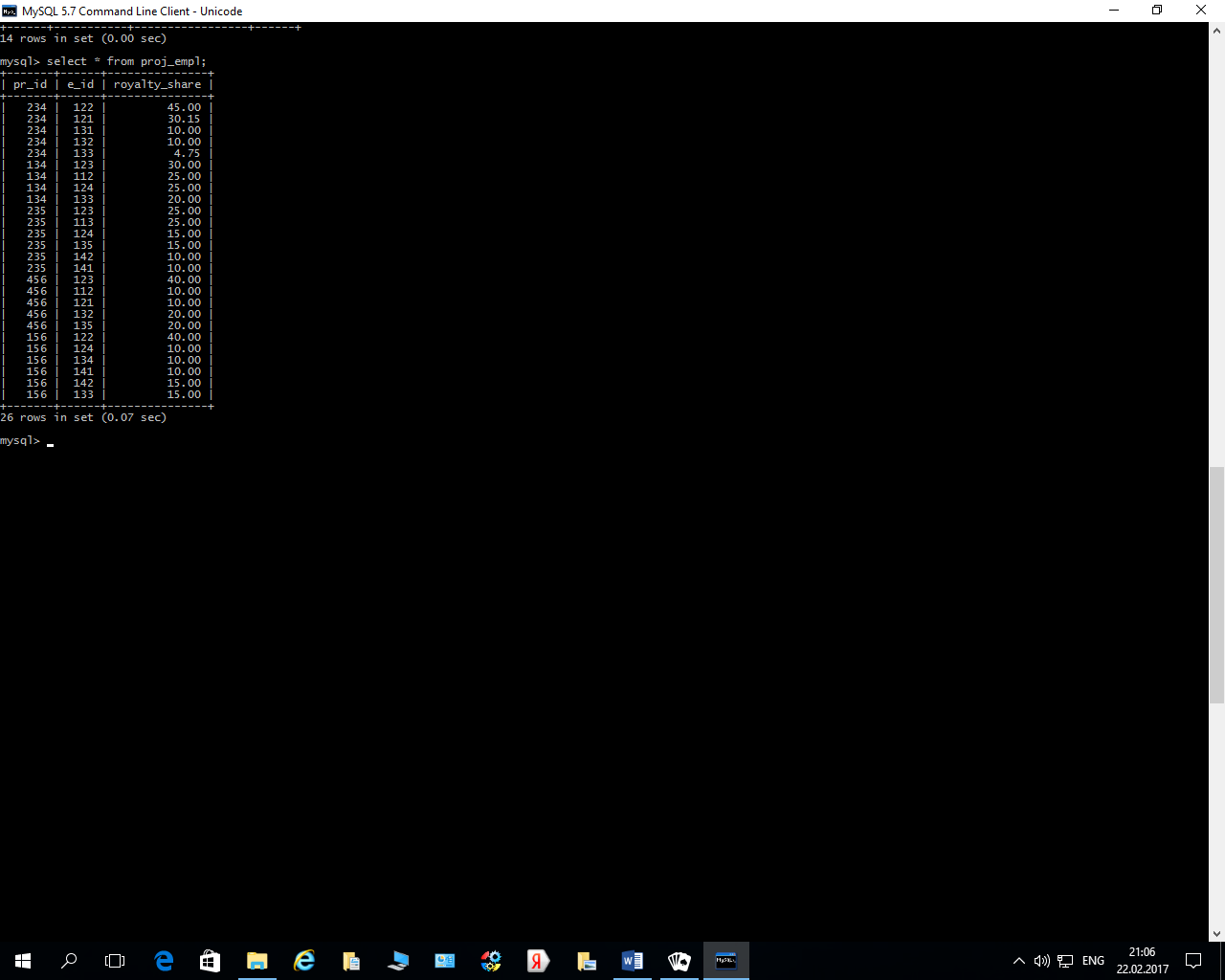
* підзапит, що повертають одне значення або рядок;
* підзапит, використовувані в логічному вираженні.

Ми розглянемо приклади підзапитів обох зазначених типів.

1.1 Підзапити похідних таблиць

Маємо наступні таблиці:





Підзапити похідних таблиць дають можливість указати запит у вираженні FROM іншого запиту. По суті це дозволяє створити тимчасову таблицю й додати її в запит. Наприклад, розглянемо наступний простий запит:

select empl\_іd, e\_name from empl where job=' Programer';

Повинно бути очевидним, що цей запит поверне імена й кодові номери всіх програмістів. Можна використати цей запит у рамках іншого, щоб одержати додаткову інформацію:

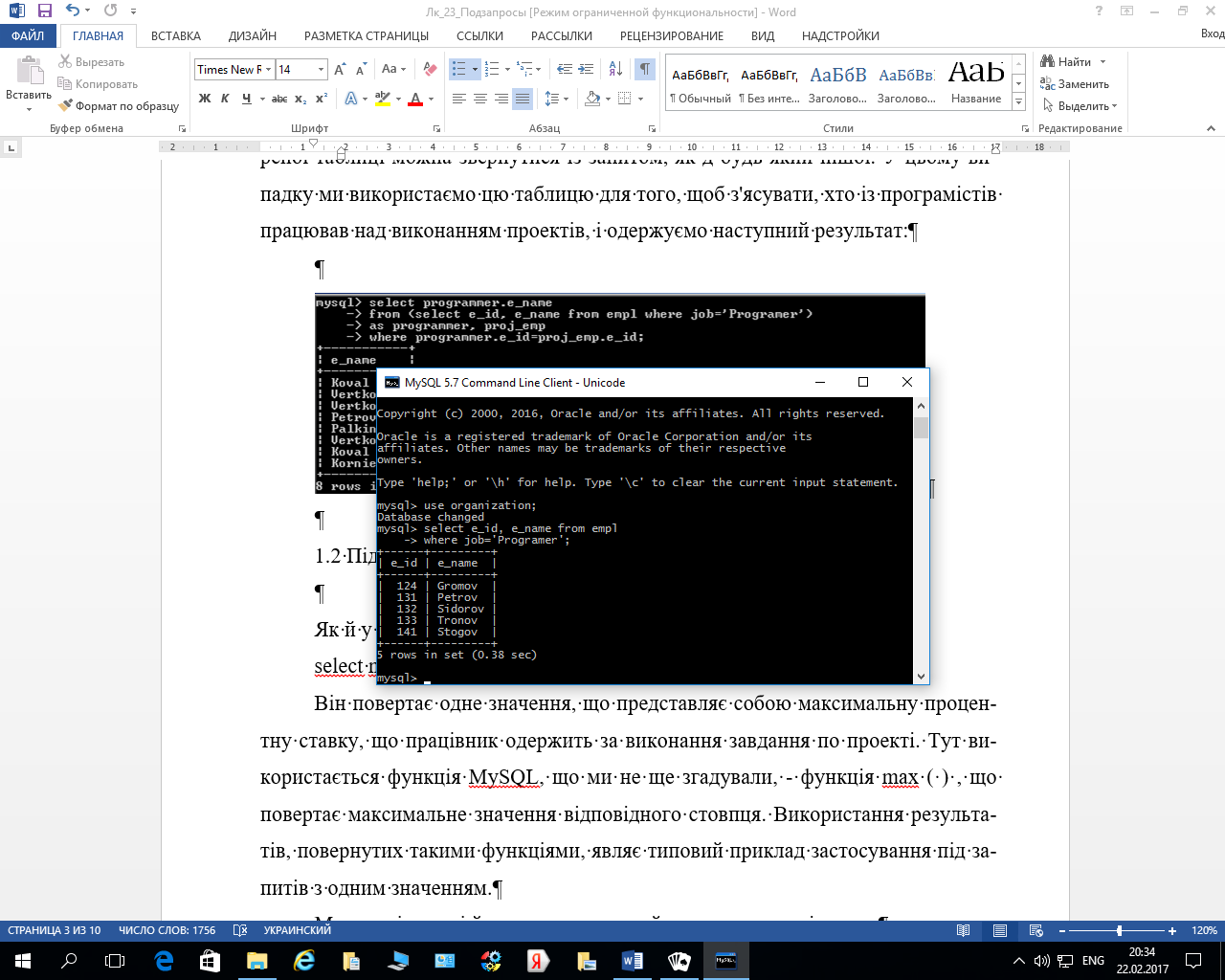
select programmer . e\_name

from (**select e\_іd, e\_name from empl where job=' Programer'**)

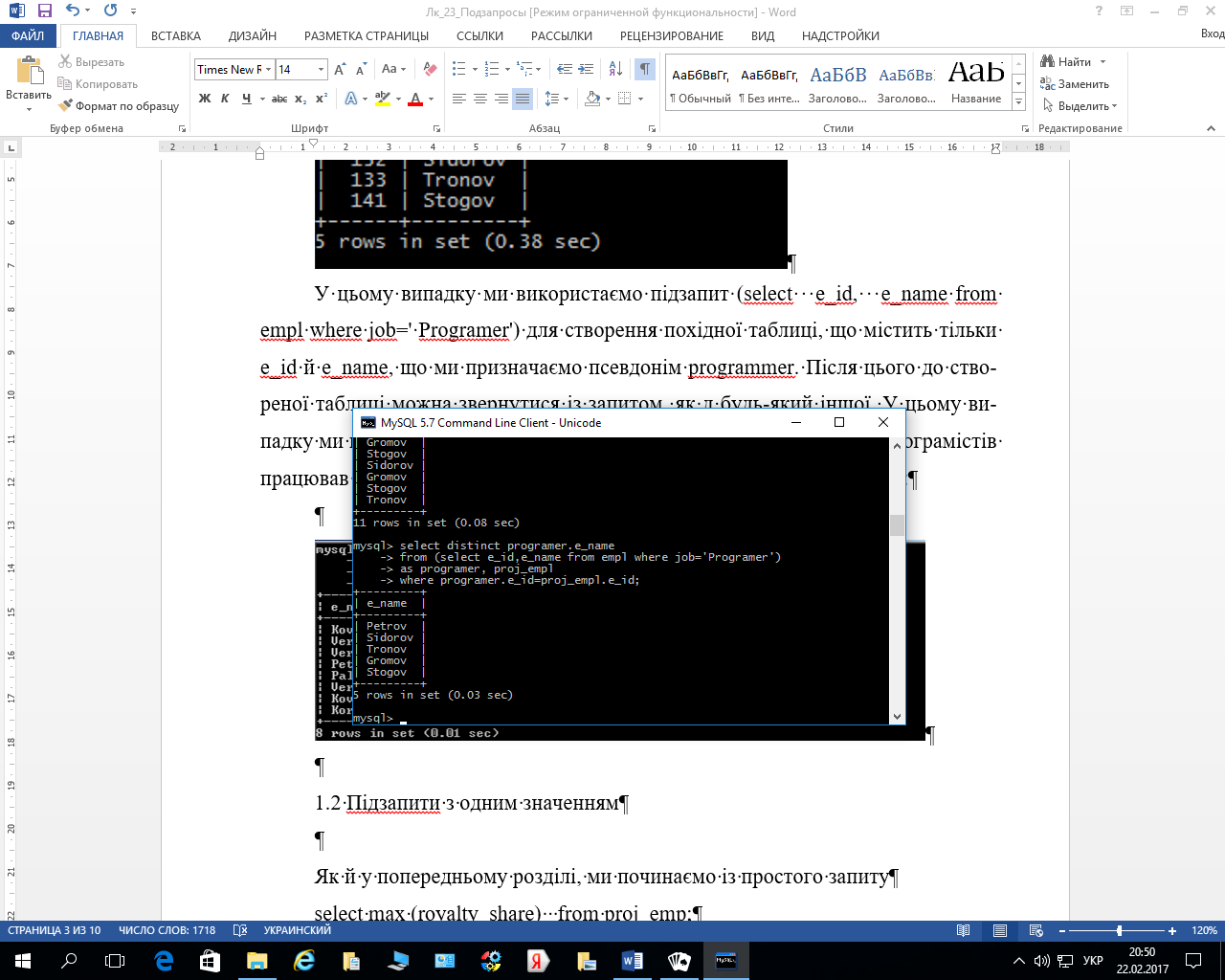
as programmer,

proj\_emp

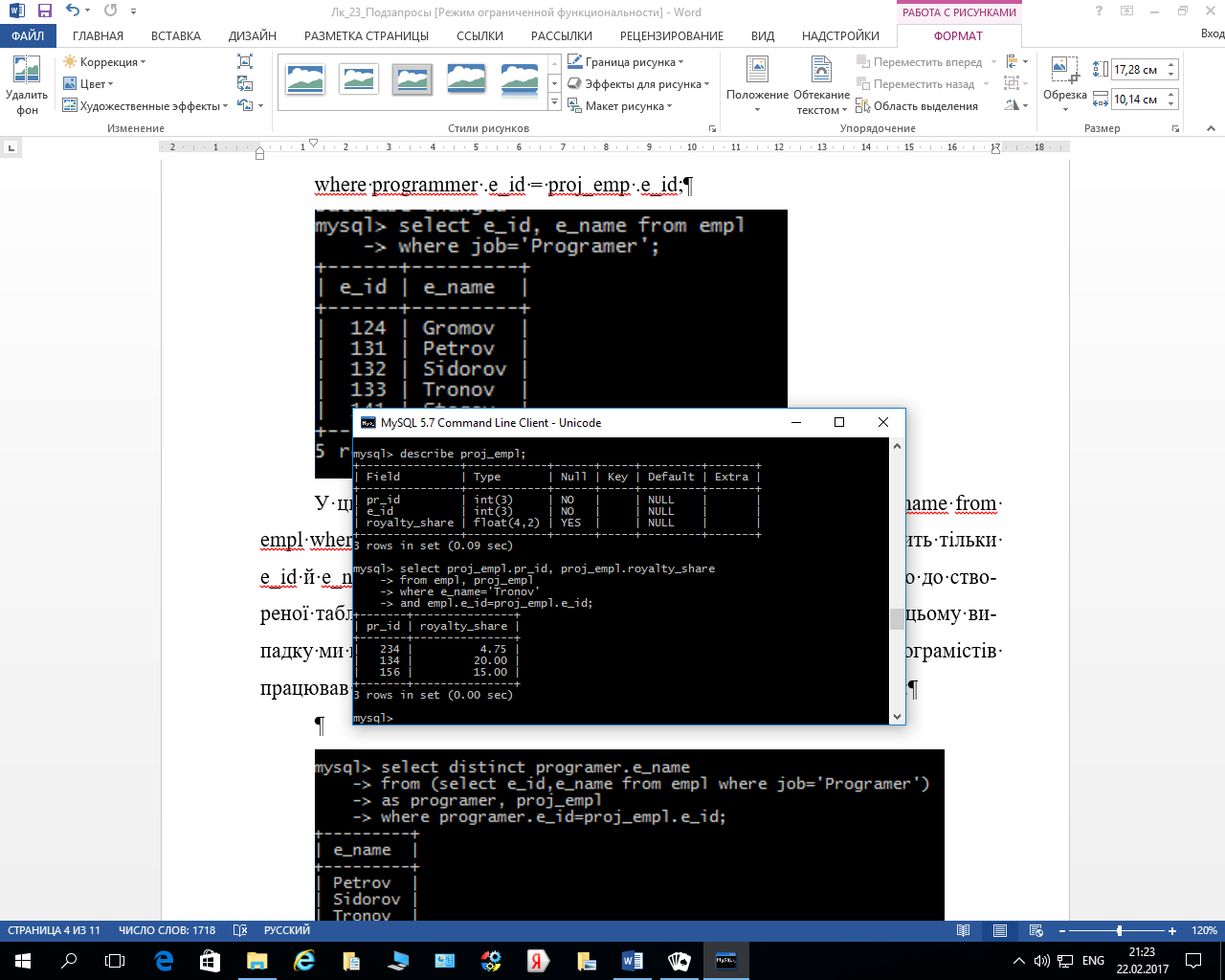
where programmer .e\_іd = proj\_emp .e\_іd;



У цьому випадку ми використаємо підзапит (select e\_іd, e\_name from empl where job=' Programer') для створення похідної таблиці, що містить тільки e\_іd й e\_name, що ми призначаємо псевдонім programmer. Після цього до створеної таблиці можна звернутися із запитом, як до будь-який іншої. У цьому випадку ми використаємо цю таблицю для того, щоб ***з'ясувати, хто із програмістів працював над виконанням проектів,*** і одержуємо наступний результат:

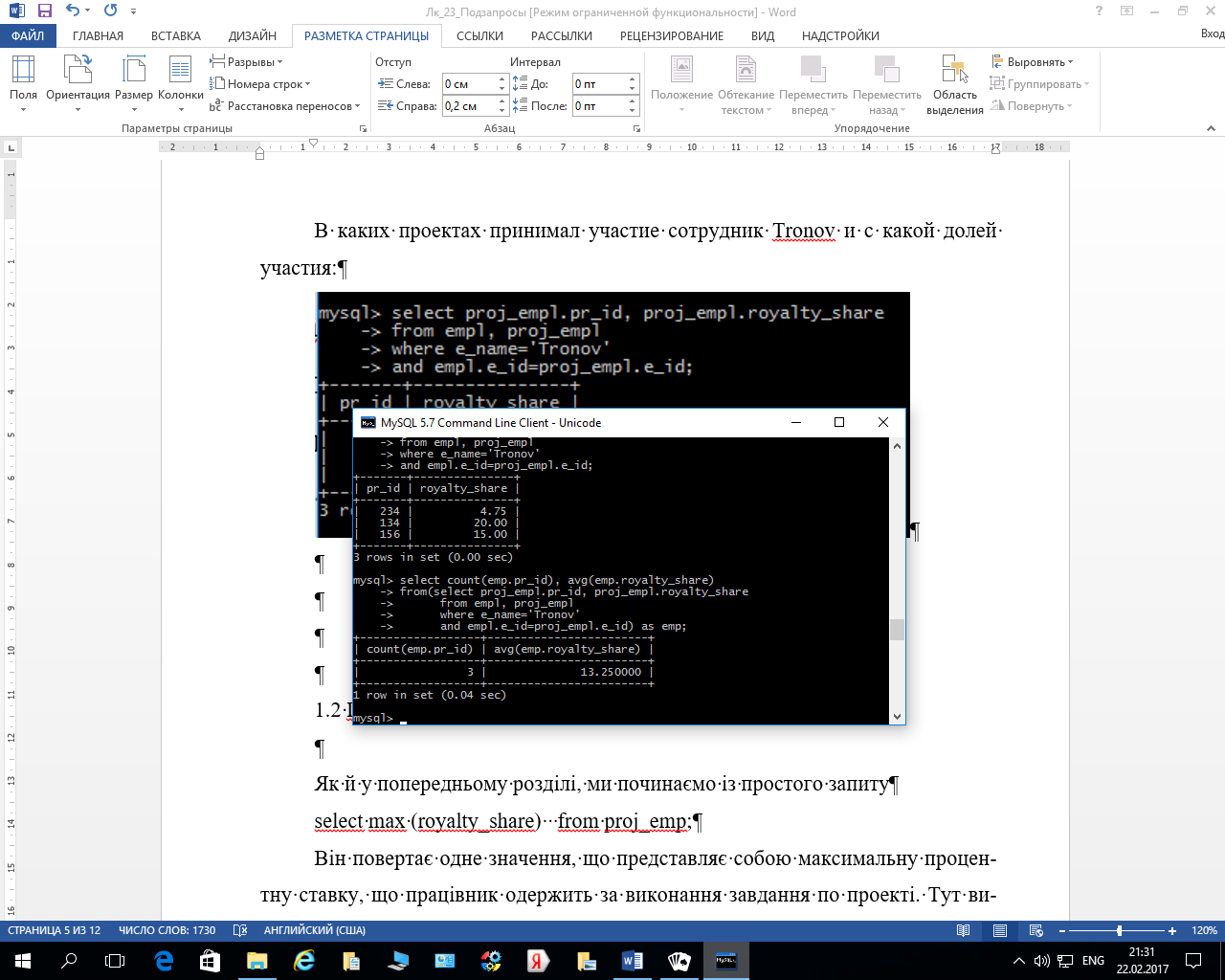


Запит***: В яких проєктах приймав участь співробітник Tronov та яка його доля участі в цих проєктах:***



Використаємо цей запит як підзапит для того, щоб отримати наступну інформацію:

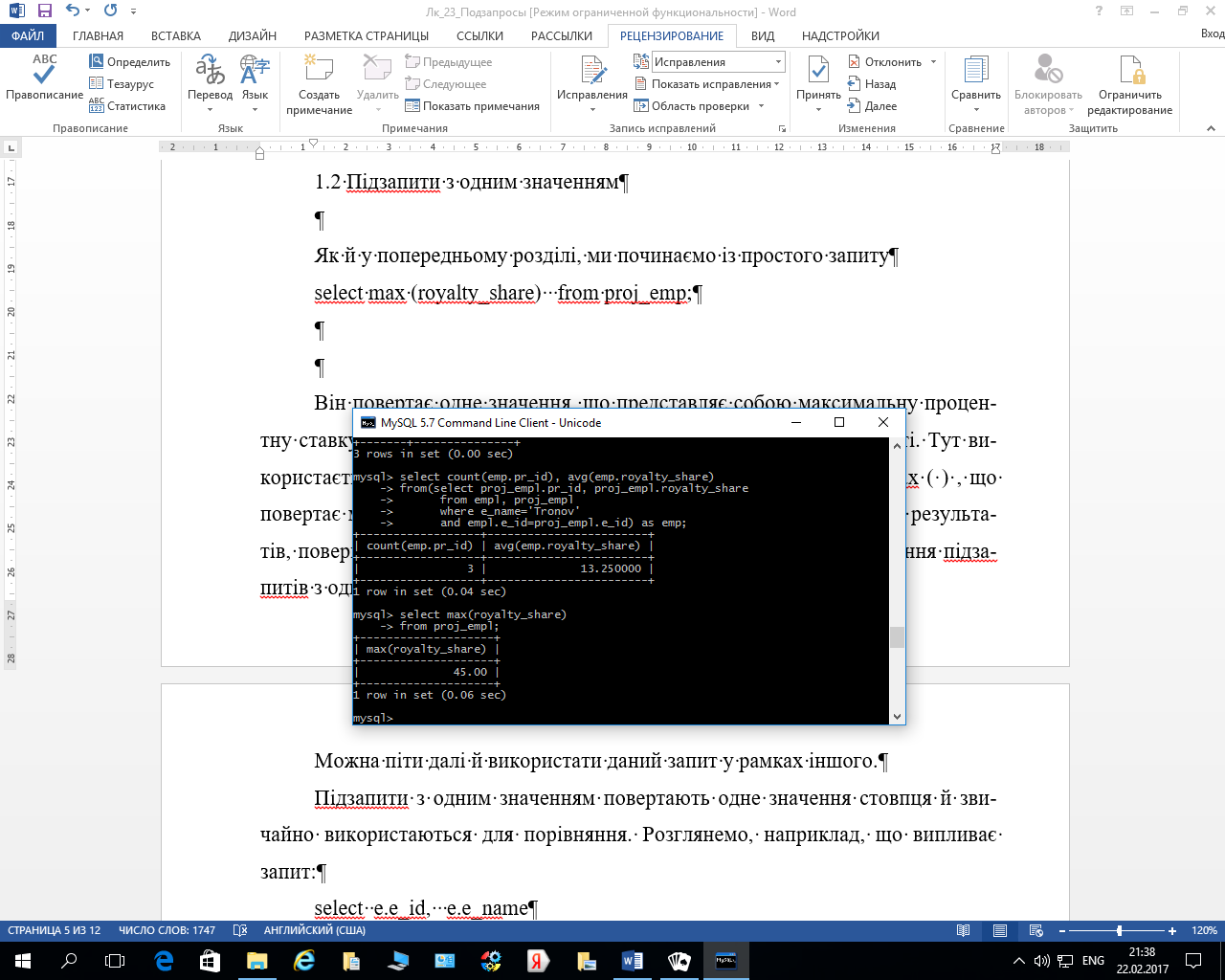
***В скількох проєктах приймав участь співробітник Tronov и яке середнє значение долі його участі в цих проєктах:***



1.2 Підзапити з одним значенням

Як й у попередньому розділі, ми починаємо із простого запиту

select max (royalty\_share) from proj\_emp;

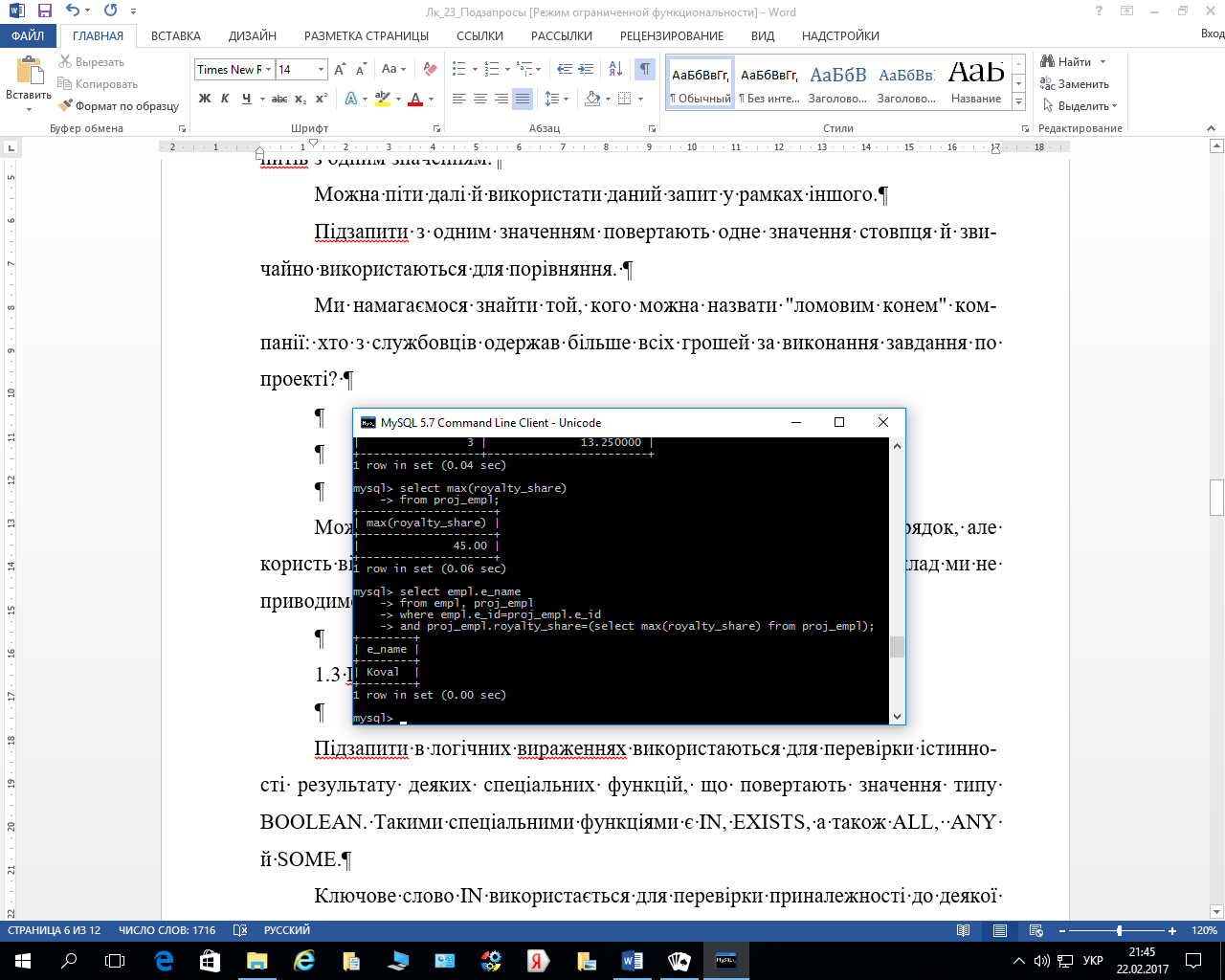


Він повертає одне значення, що представляє собою максимальну процентну ставку, яку працівник отримає за виконання завдання по проекті. Тут використається функція MySQL, що ми не ще згадували, - функція max ( ) , що повертає максимальне значення відповідного стовпця. Використання результатів, повернутих такими функціями, являє типовий приклад застосування підзапитів з одним значенням.

Можна піти далі й використати даний запит у рамках іншого.

Підзапити з одним значенням повертають одне значення стовпця й звичайно використаються для порівняння.

Ми намагаємося знайти той, кого можна назвати "ломовим конем" компанії: хто з службовців одержав більше всіх грошей за виконання завдання по проекті?



Можна також створити підзапит, що повертає не значення, а рядок, але користь від такого підзапита досить обмежена, тому відповідний приклад ми не приводимо.

1.3 Підзапити в логічних виразах

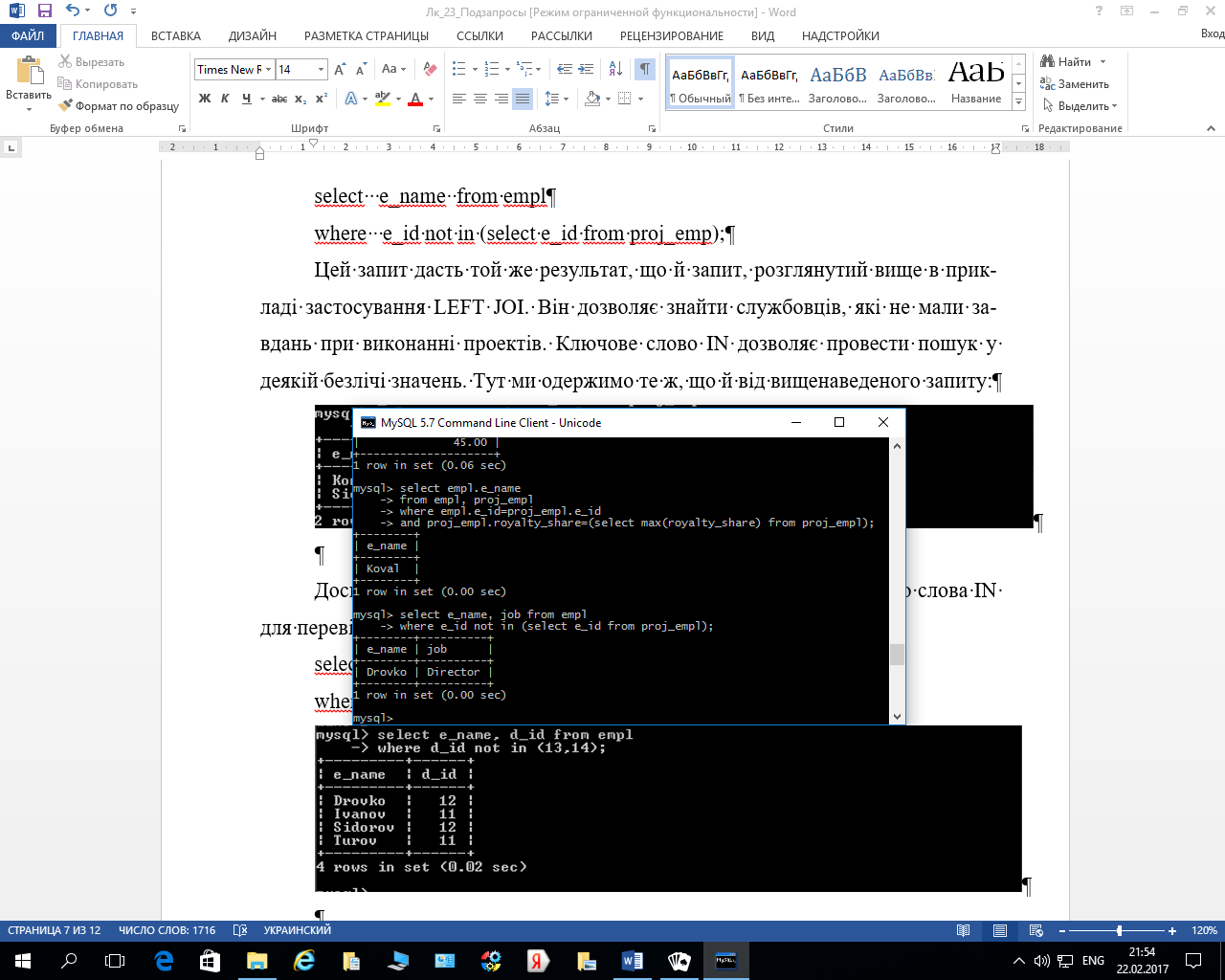
Підзапити в логічних виразах використаються для перевірки істинності результату деяких спеціальних функцій, що повертають значення типу BOOLEAN. Такими спеціальними функціями є ІN, EXІSTS, а також ALL, ANY й SOME.

Ключове слово ІN використається для перевірки приналежності до деякої множини значень. Розглянемо наступний запит:

select e\_name from empl

where e\_іd not іn (select e\_іd from proj\_emp);

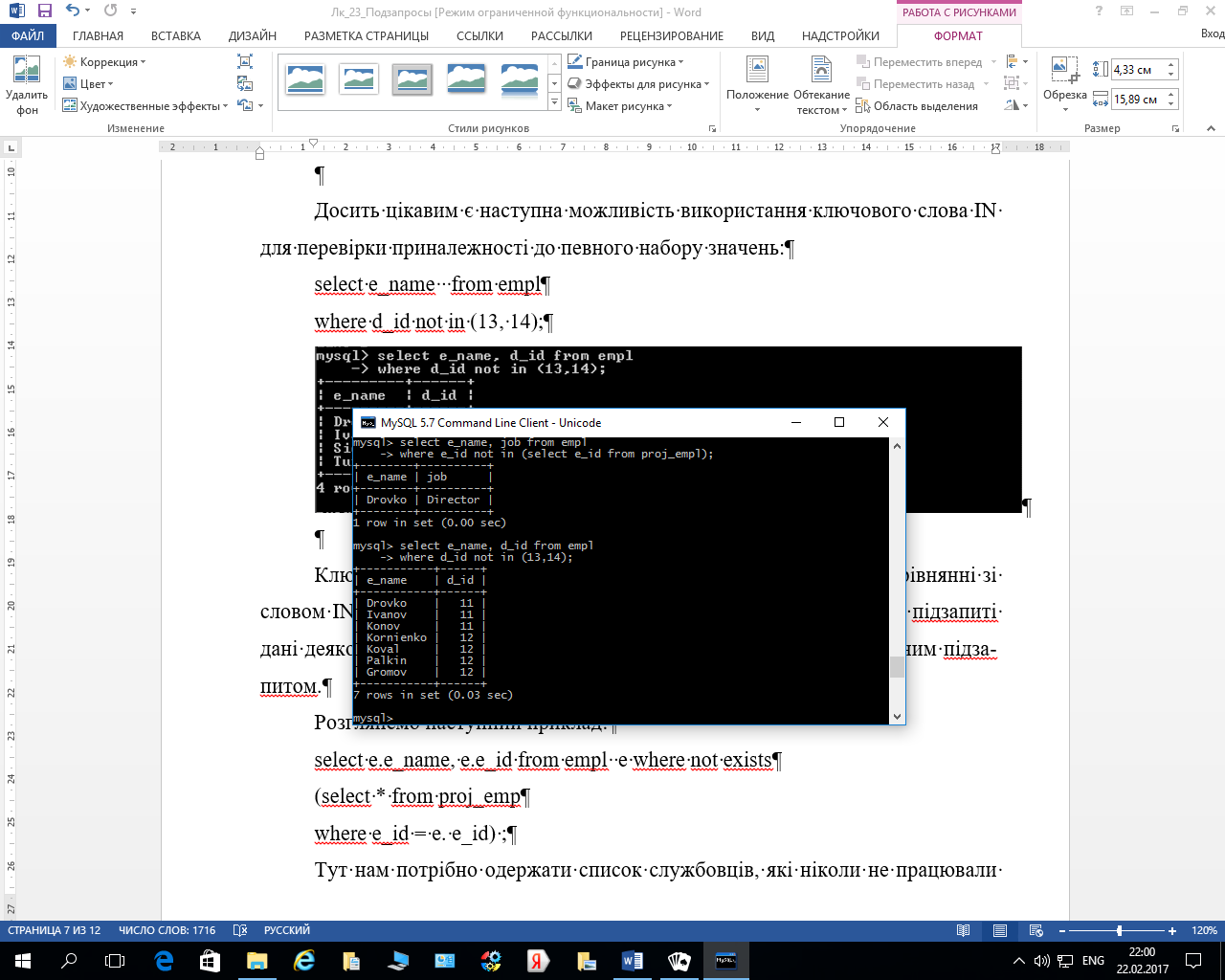
Цей запит дасть той же результат, що й запит, розглянутий вище в прикладі застосування LEFT JOІ. Він дозволяє знайти службовців, які не мали завдань при виконанні проектів. Ключове слово ІN дозволяє провести пошук у деякій множині значень. Тут ми одержимо те ж, що й від вищенаведеного запиту:



Досить цікавим є наступна можливість використання ключового слова ІN для перевірки приналежності до певного набору значень:

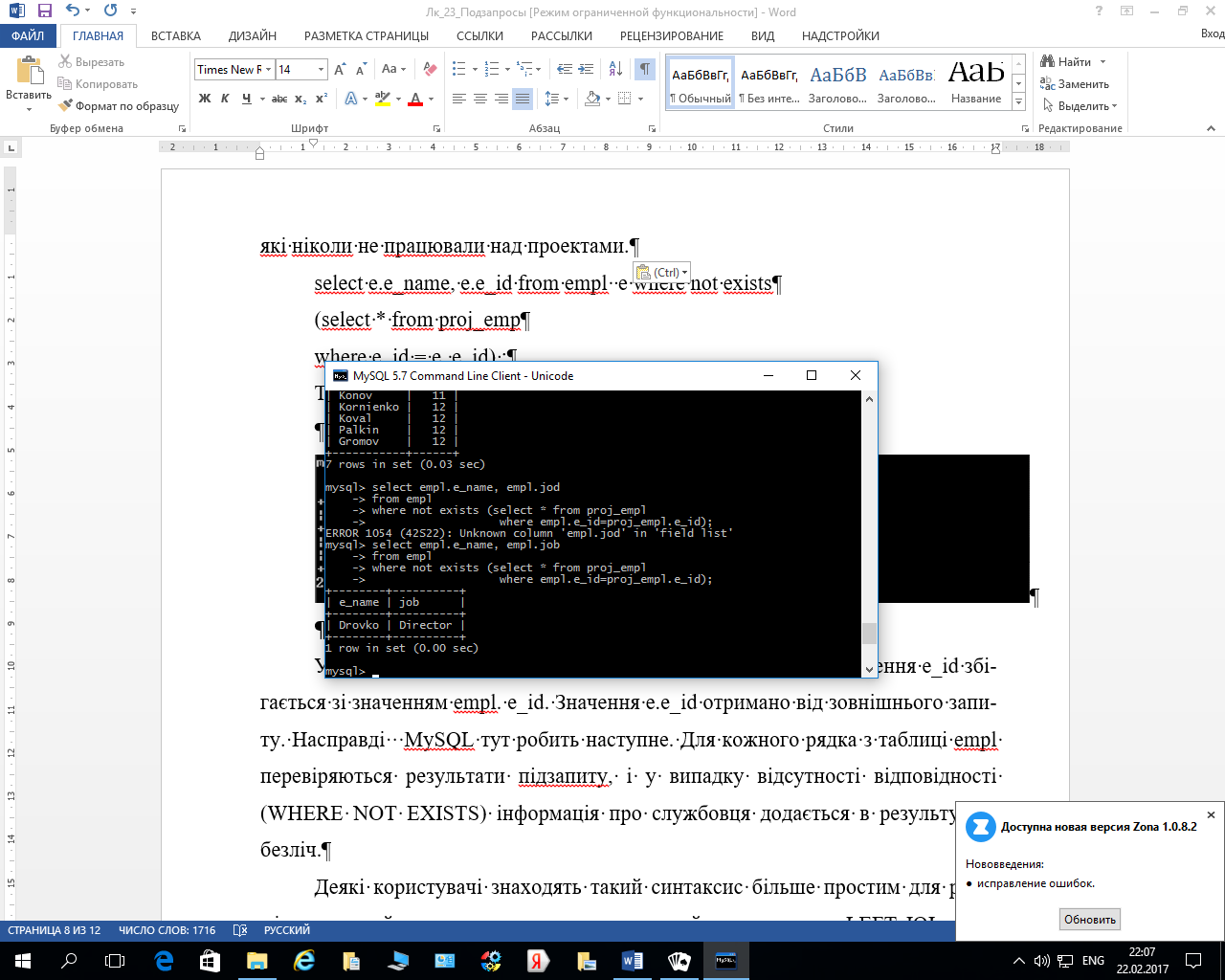
select e\_name from empl

where d\_іd not іn (13, 14);



Ключове слово EXІSTS працює лише небагато по-іншому в порівнянні зі словом ІN. При використанні EXІSTS ми насправді використаємо в підзапиті дані деякого зовнішнього запиту. Такий запит іноді називають зв'язаним підзапитом.

Розглянемо наступний приклад: потрібно одержати список службовців, які ніколи не працювали над проектами.

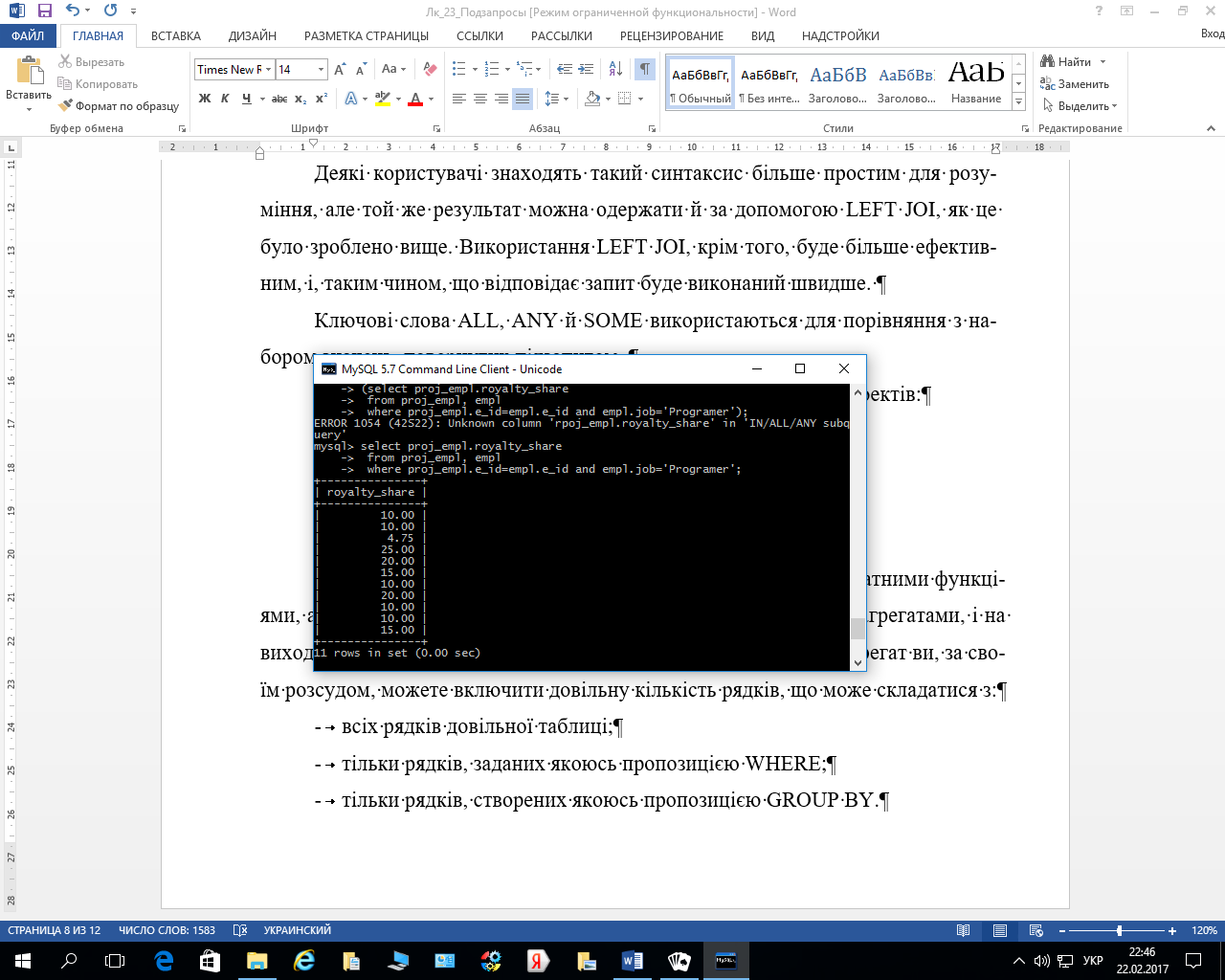


У підзапиті ми шукаємо рядки таблиці proj\_emp, у яких значення e\_іd збігається зі значенням empl. e\_іd. Значення empl.e\_іd отримано від зовнішнього запиту. Насправді MySQL тут робить наступне. Для кожного рядка з таблиці empl перевіряються результати підзапиту, і у випадку відсутності відповідності (WHERE NOT EXІSTS) інформація про службовця додається в результуючу безліч.

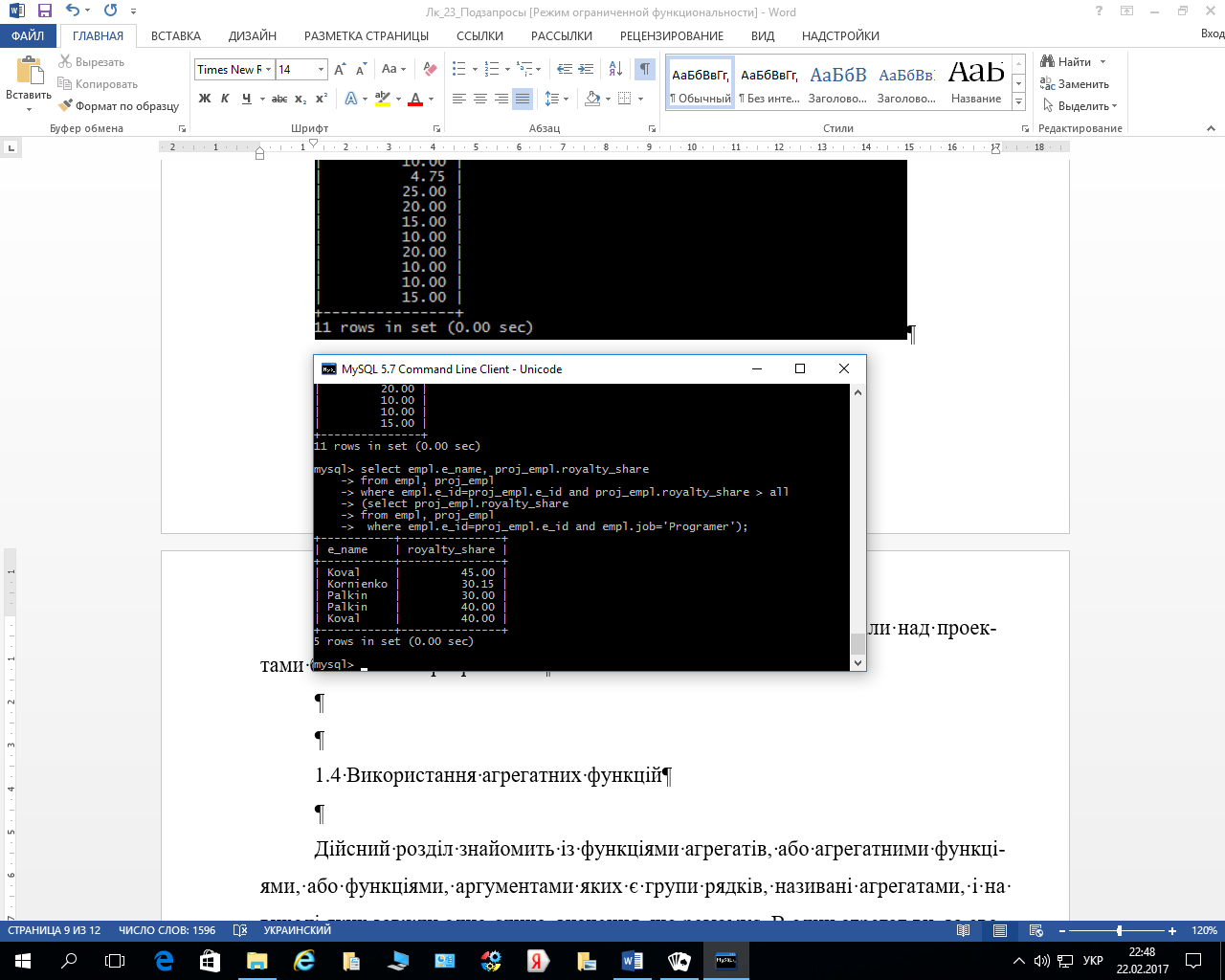
Деякі користувачі знаходять такий синтаксис більше простим для розуміння, але той же результат можна одержати й за допомогою LEFT JOІ, як це було зроблено вище. Використання LEFT JOІ, крім того, буде більше ефективним, і, таким чином, що відповідає запит буде виконаний швидше.

Ключові слова ALL, ANY й SOME використаються для порівняння з набором значень, повернутих підзапитом.

Отримаємо спочатку долі участі програмістів в розробці проектів:



А тепер отримаємо список тих співробітників, які працювали над проектами більше всіх програмістів:



1.4 Використання агрегатних функцій

Дійсний розділ знайомить із функціями агрегатів, або агрегатними функціями, або функціями, аргументами яких є групи рядків, називані агрегатами, і на виході яких завжди одне-єдине значення, що резюмує. В один агрегат ви, за своїм розсудом, можете включити довільну кількість рядків, що може складатися з:

* всіх рядків довільної таблиці;
* тільки рядків, заданих якоюсь пропозицією WHERE;
* тільки рядків, створених якоюсь пропозицією GROUP BY.

Незалежно від того, скільки саме рядків ви включите у свій агрегат, агрегатна функція видасть для нього тільки одне значення, наприклад статистичне (сума, мінімум або середнє).

Функція Результат

MІN(expr) Мінімальне значення в expr

MAX(expr) Максимальне значення в expr

SUM(expr) Сума всіх значень в expr

AVG(expr) Середнє всіх значень в expr

COUNT(expr) Число всіх значень, що не є значеннями null, в expr

COUNT(\*) Число рядків у довільній таблиці або довільному наборі рядків

Можна також створити підзапит, що повертає не значення, а рядок, але користь від такого підзапита досить обмежена, тому відповідний приклад ми не приводимо.

ЗМІСТ І ПОСЛІДОВНІСТЬ ВИКОНАННЯ ЗАВДАНЬ

1. Реалізувати за допомогою оператора SELECT 4 запити:

1 Підзапити похідних таблиць

2 Підзапити з одним значенням

3 Підзапити в логічних вираженнях

4 Використання агрегатних функцій

ІІ. Оформити звіт по виконанню лабораторної роботи. Звіт повинен включати наступні розділи для кожного запиту:

1. Формулювання запиту.

2. Виведення вмісту таблиці, для якої сформований запит.

3. Оператор SELECT до відповідного запиту.

4. Копія вікна виконання оператора SELECT.