УДК 004.658
DOI https://doi.org/10.32782/IT/2021-2-2

Леонід КАБАК
кандидат технічних наук, доцент, доцент кафедри програмного забезпечення комп'ютерних систем, Національний технічний університет «Дніпровська політехніка», просп. Дмитра Яворницького, 19, м. Дніпро, Україна, 49005, kabak.leo@gmail.com
***ORCID:** 0000-0001-6267-1772*
***Scopus Author ID:** 57202222055*

Борис МОРОЗ
доктор технічних наук, професор, професор кафедри програмного забезпечення комп'ютерних систем, Національний технічний університет «Дніпровська політехніка», просп. Дмитра Яворницького, 19, м. Дніпро, Україна, 49005, moroz.boris.1948@gmail.com
***ORCID:** 0000-0002-5625-0864*
***Scopus Author ID:** 57218242332*

Валерій КОВАЛЬ
студент спеціальності 121 «Інженерія програмного забезпечення» другого (магістерського) рівня вищої освіти, Національний технічний університет «Дніпровська політехніка», просп. Дмитра Яворницького, 19, м. Дніпро, Україна, 49005

Бібліографічний опис статті: Кабак, Л., Мороз, Б., Коваль, В. (2021). Метод та алгоритм уникнення фрагментації індексів в базах даних. *Information Technology: Computer Science, Software Engineering and Cyber Security*, 2, 10–18, doi: https://doi.org/10.32782/IT/2021-2-2

МЕТОД ТА АЛГОРИТМ УНИКНЕННЯ ФРАГМЕНТАЦІЇ ІНДЕКСІВ У БАЗАХ ДАНИХ

У роботі було розглянуто фрагментацію індексів, проаналізовані фактори зменшення продуктивності виконання SQL запитів при збільшенні відсотка фрагментації індексів. В роботі проаналізовано систему індексації, яка використовується в СУБД ORACLE. Розглянуті і проаналізовані сучасні методи зменшення фрагментації та розроблено модель, методи та алгоритм який в автоматичному режимі дозволить робити перебудову та реорганізацію індексів в залежності від відсотка фрагментації. ***Метою роботи*** *є розробка метода та алгоритмів уникнення фрагментації індексів. Реалізація поставленої мети передбачає вирішення завдання пошуку аналізу та перебудові фрагментованих індексів, що забезпечує запропонований у роботі метод. На базі запропонованого методу було розроблено алгоритм для систем баз даних, та розроблено процедуру яка дозволяє уникати фрагментації індексів у СУБД ORACLE, які відбуваються при роботі серверу ORACLE у режимі OLTP.* ***Методологія*** *вирішення поставленого завдання полягає в проведенні статистичного аналізу існуючих індексів та розробці системи яка дозволяє в запланований час, в який сервер баз даних менш всього навантажений запускати процедуру яка буде перевіряти ступінь фрагментації індексів і при потребі їх перебудовувати.* ***Наукова новизна****. В ході виконання роботи набув подальший розвиток метод перебудови індексів, які фрагментовано, у базі даних. В перше запропоновано використання перебудови індексів у автоматичному режимі без втручання адміністратора бази даних.* ***Висновки****. Результати даної роботи можуть бути використані для подальших досліджень і розробок, а також для запровадження використання технології уникнення фрагментації для різних типів СУБД. Всі отримані результати представлені в графічному вигляді з детальним описом в даній роботі.*

Ключові слова: *база даних Oracle, індекси, OLTP сервер, виявлення фрагментації, перебудова індексів.*

Leonid KABAK
Candidate of Technical Sciences, Associate Professor of Department of Software Engineering, Dnipro University of Technology, 19 Dmytra Yavornytskoho ave., Dnipro, Ukraine, 49000, kabak.leo@gmail.com
ORCID: *0000-0001-6267-1772*
Scopus-Author ID: *57202222055*

Borys MOROZ
Doctor of Technical Sciences, Professor of Department of Software Engineering, Dnipro University of Technology, 19 Dmytra Yavornytskoho ave., Dnipro, Ukraine, 49000, moroz.boris.1948@gmail.com
ORCID: *0000-0002-5625-0864*
Scopus-Author ID: *57202222055*

Valerii KOVAL
Student of specialty 121 "Software Engineering" of the second (master's) level of higher education, Dnipro University of Technology, 19 Dmytra Yavornytskoho ave., Dnipro, Ukraine, 49005

To cite this article: Kabak, L., Moroz, B., Koval, V. (2021). Metod ta alhorytm unyknennia frahmentatsii indeksiv u bazakh danykh [Method and algorithm for avoiding index fragmentation in database]. *Information Technology: Computer Science, Software Engineering and Cyber Security*, 2, 10–18, doi: https://doi.org/10.32782/IT/2021-2-2

METHOD AND ALGORITHM FOR AVOIDING INDEX FRAGMENTATION IN DATABASE

The paper considers index fragmentation, analyzes the decrease in the performance of SQL queries with increasing percentage of index fragmentation. The indexing system used in the ORACLE database is analyzed in the paper. Modern methods of fragmentation reduction are considered and analyzed and a model, methods and algorithm are developed which in automatic mode will allow to make reorganization and reorganization of indices depending on the percentage of fragmentation. ***The aim*** *of the work is to develop a method and algorithms to avoid index fragmentation. The realization of this goal involves solving the problem of finding analysis and restructuring of fragmented indices, which provides the proposed method. Based on the proposed method, an algorithm for database systems was developed, and a procedure was developed to avoid index fragmentation in the ORACLE database, which occurs when the ORACLE server is running in OLTP mode.* ***Scientific novelty****. In the course of the work, the method of rearranging the fragmented indices in the database was further developed. In the first, it is proposed to use index restructuring in automatic mode without the intervention of the database administrator.* ***Conclusions.*** *The information system was built in which the method was developed and the algorithm for the system of avoiding index fragmentation in ORACLE DBMS, which occur during the operation of the ORACLE server in OLTP mode, was proposed. The results are represented graphically and described in this work as well.*

Key words: *Oracle database, indexes, OLTP server, fragmentation detection, index restructuring.*

Актуальність проблеми. Нині в банківських інформаційних системах та в інформаційних системах різних підприємств повсюдно використовують сервер баз даних, для збереження та обробки даних. Основним інструментом підвищення швидкості доступу до даних для додатків та користувачів бази даних, звичайно є індекси, кластери, хеш-кластери та механізми секціонування. Сервери що працюють у цих організаціях працюють у режимі багато навантажених OLTP (Online Transaction Processing) серверів, тому операції вставки та оновлення даних досить часті. До даних що знаходяться в цих базах даних використовуються різні операції вставки, зміни та видалення ці операції приводять до того, що стовпці по яких були проіндексовані таблиці змінились чи видалились, а запис в індексі не змінився таке явище називається фрагментацією. Фрагментація має місце в тих випадках, коли в індексах містяться сторінки, для яких логічний порядок, заснований на значенні ключа, не збігається з фізичним порядком у файлі даних. Якщо фрагментованих індексів багато то це дуже сильно уповільнить доступ до даних і швидкодія роботи серверів баз даних та додатків значно знизиться [1].

Процесом відстеження фрагментації індексів займається адміністратор бази даних однак в базі даних таких об’єктів як таблиці дуже забагато та дуже забагато індексів і адміністратор бази даних не встпіває відслідковувати правильність налаштування індексів. Тому у роботі пропонується розробка алгоритму та методу для відстеження фрагментації індексів та автоматичного запуску процедури перебудови чи реорганізації індексів в залежності від проценту фрагментації в час коли сервер не перевантажений.

У роботі було проаналізовано роботу системи індексації на прикладі СУБД Oracle. Розроблено методи та модель системи яка буде дозволяти проводити аналіз існуючих індексів та приймати рішення по їх перебудову та реорганізацію. Розглянуто можливість реалізації даної системи в будь якій установі та на підприємстві де використовуються OLTP сервери баз даних.

Аналіз останніх досліджень і публікацій. На цей час у базах даних накопичені сотні терабайтів інформації. Для того щоб знаходити необхідні данні потрібно оперувати з величезними масивами даних. З цими даними одночасно оперують багато користувачів, задача сучасних СУБД надати користувачам як змога швидше необхідні данні. Для прискорення пошуку необхідних даних в СУБД використовуються наступні механізми, а саме індексування, кластерізації, секціонування та хешування. Но все ж таки на цей час головним інструментом прискорення доступу до даних залишається індексування. Індекс є не обов'язковою структурою збереження даних і використовується тільки в тих випадках коли потрібно прискорити пошук необхідних даних за певними значеннями, як правило індексуються тільки ті стовпці по значенням яких найчастіше проводиться вибірка даних [1; 2].

Стовпці таблиць які потрібно індексувати, як правило призначає розробник додатку, однак в процесі експлуатації СУБД адміністратор може сам добавляти, або видаляти індекси за потребою, після проведення налаштування продуктивності роботи запитів до бази даних [3].

В індексах данні зберігаються упорядковано або за зростанням, в індексі зберігається вказівник на фізичне місце зберігання даних в СУБД Oracle воно називається ROWID. Якщо БД працює у режимі навантаженого серверу то операції оновлення та видалення даних відбуваються доволі часто. У роботах [4–6] були описані проблеми які часто виникають при роботі з індексами:

– Відсутність індексу по стовпцю по якому часто проводиться вибірка;

– Не ефективна побудова індексів;

– Індекси блокуються операціями конкатенації чи вибірками з використанням агрегатних функцій;

– Індекси конкурують між собою;

– Індекс має великий фактор кластеризації CLUSTERING_FACTOR;

– Індекс сильно фрагментований.

В статі буде розглянуто саме вирішення проблеми фрагментації індексів. Дослідженням технологій використання індексів у базах даних займались також вітчизняні вченні. У навчальних посібниках [9; 10] були наведені приклади та описана технологія використання індексів в сучасних СУБД. Виклад матеріалу у цих посібниках було зосереджено на задачах підвищення продуктивності роботи сервера за рахунок проведення індексації таблиць БД.

Мета статті. На підставі дослідженого та проаналізованого матеріалу наукових досліджень було запропоновано метод та модель для запобіганню фрагментації індексів за рахунок їх перебудови та реорганізації. В статі запропоновано метод запобігання фрагментації. Завдяки наведеному методу було розроблено алгоритм та його програмна реалізація на мові PL/SQL для запобіганню процесу фрагментації. Розглядаються можливості використання розробленого методу та його подальша інтеграція до банківських систем та систем в яких використовується СУБД Oracle. Основна мета статі полягає в описі розробленого методу та опису можливостей його подальшого використання в різних інформаційних системах. Для реалізації цієї задачі в статі викладено побудову методу та наведено модель системи запобіганню фрагментації даних.

Виклад основного матеріалу. При розробці додатків для роботи з базами даних для прискорення пошуку даних використовують індекси. Занадто багато індексів у додатку прискорює операції вибірки даних з БД, занадто мало індексів пригальмовує виконання операцій DML операцій таких як вставка та оновлення даних. Доволі часто спочатку розробляється додаток а потім визначаються стовпці які потребують індексації. Це означає що потрібно витратити деякий час для пошуку в таблицях стовпців за значеннями яких найчастіше проводяться вибірки даних. Якщо деякі індекси ніколи не використовуються чи використовуються доволі рідко їх потрібно видалити для того щоб вони не заважали роботі додатку.

Розробник додатків повинен турбуватись про створення індексів, які використовуються в його додатках, а адміністратор БД повинен відстежувати зростання індексів їх фізичного розміщення з метою прискорення швидкодії роботи БД.

В СУБД ORACLE використовуються наступні види індексів.

1. Індекси B-дерева. Цей індекс використовуються в Oracle та більшості інших баз даних. Індекси B*Tree забезпечують швидкий доступ за допомогою ключа до певного рядка або діапазону рядків. Для пошука потрібних рядків

потрібно декілька операцій читання, щоб знайти саме той рядок, що потрібен [2].

2. Індекси бітових карт. Бітовий індекс, використовує бітовий малюнок, щоб одночасно вказувати на багато рядків. Вони підходить для даних які часто повторюються (даних з декількома різними значеннями відносно загальна кількість рядків у таблиці), що в основному використовується для читання [2].

Як показано на рис. 1 та рис. 2 індекс знаходить ROWID запису і виходячі зі значення цього ROWID робить фізичне читання цього запису з вказанного у ROWID місця. Наприклад:

ROWID: BBBBBBBBB.RRRR.FFFF

BBBBBBBBBB – шістнадцятирічний номер блоку в файлі даних, в якому знаходиться рядок;

RRRR – шістнадцятирічний номер рядка в блоці;

FFFF – файл, в состав якого входе цей блок.

Починаючи з 10 версії Oracle ROWID у БД зберігається у зашифрованому вигляді і щоб

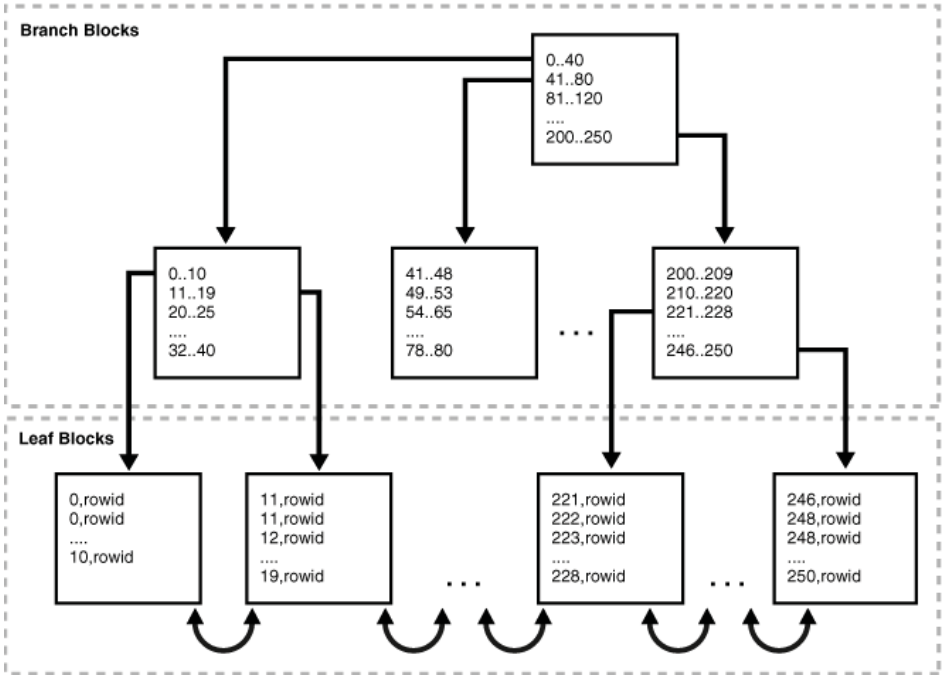

Рис. 1. Індекси B-дерева [2]

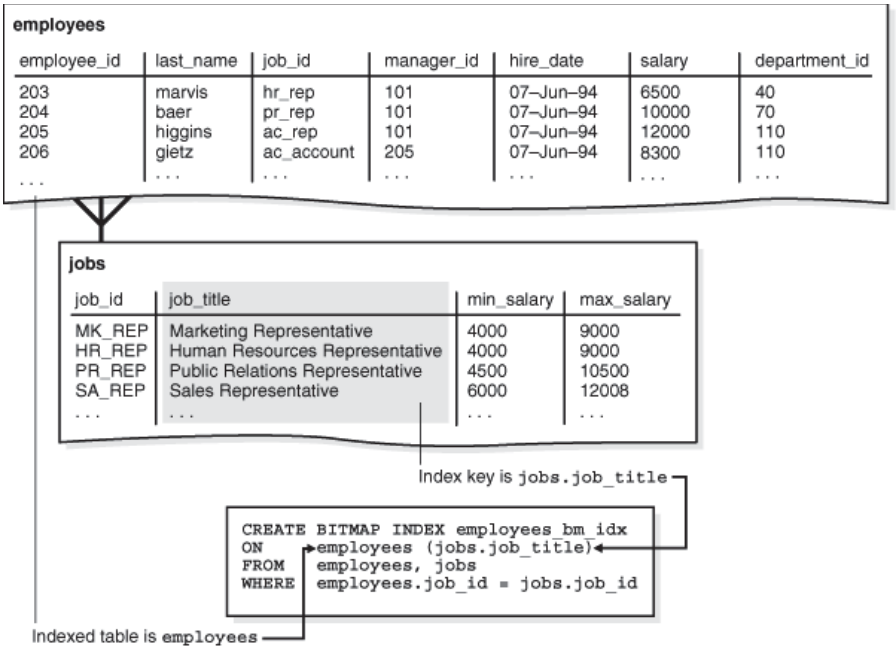

Рис. 2. Індекси бітових карт [2]

переглянути значення ROWID використовують спеціальний скріпт рис. 3.

```
   select rowid as therowid, d.department_id,
   to_number(utl_encode.base64_decode(utl_
raw.cast_to_raw(lpad(substr(rowid,1, 6), 8,
'A'))), 'XXXXXXXXXXXX') as objid,
   to_number(utl_encode.base64_decode(utl_
raw.cast_to_raw(lpad(substr(rowid, 7, 3), 4,
'A'))), 'XXXXXX') as filenum,
   to_number(utl_encode.base64_decode(utl_
raw.cast_to_raw(lpad(substr(rowid, 10, 6), 8,
'A'))), 'XXXXXXXXXXXX') as blocknum,
   to_number(utl_encode.base64_decode(utl_
raw.cast_to_raw(lpad(substr(rowid, 16, 3),
4, 'A'))), 'XXXXXX') as rowslot   from demo.
department d
```

Коли БД працює у високо навантаженому режимі то явище фрагментації індексів зустрічається доволі часто. Якщо індекс фрагментований то значення ROWID не співпадає з фізичним знаходженням записів. Тоді для пошуку інформації потрібно буде робити набагато більше операцій читання, чим при використанні індексу. В цьому випадку всі додатки яки шукають данні використовуючи цей індекс будуть працювати повільно. Цей недолік є майже у всіх базах даних. Так корпорація Microsoft рекомендує, якщо індекс фрагментовано більше ніж на 5% та менше ніж на 30% то його потрібно реорганізувати, якщо більше ніж 30% то індекс потрібно перебудувати [1]. За видами фрагментацію розрізняють на внутрішньоблокову фрагментацію, яка виникає в табличних екстентах, внутрішньоблокову яка виникає в індексних екстентах та міжблочну фрагментацію.

Внутрішньоблокова фрагментація, яка виникає в табличних екстентах виникає тоді коли з таблиці видаляється багато рядків. Тоді сповільнюється багато вільної пам'яті у блоках даних в яких зберігалась таблиця. Но нові операції вставки не використовують старі блоки даних а вставляють данні в нові блоки які розташовані уже в інших екстентах [5–7].

Внутрішньоблокову яка виникає в індексних екстентах виникає тоді коли з таблиць видаляється багато рядків вони і вони в одночас видаляються і з індексних блоків. Якщо до цієї таблиці звертається багато транзакцій, які модифікують чи видаляють рядки то індекси ціє таблиці дуже швидко набирають свій обсяг. Особливо це відбувається при використанні BitMap індексу та секціонованного індексу. Якщо у таблиці мається велика кількість індексів це може викликати серіозну проблему стосовно швидкодії транзакій яки звертаються до цієї таблиці [8].

Міжблочна фрагментація, яка виникає тоді коли модифікуються данні в таблиці та в блоках бази даних в яких зберігались рядки цієї таблиці не вистачає місця для збереження модифікованих рядків і вони переносяться в інші блоки, а блоки де вони зберігались становляться повністю спустошеними. Наявність таких блоків уповільнює швидкість сканування таблиці для пошуку даних навіть з використанням індексів.

Для відстеження роботи індексу використовується команда ANALIZE [назва індексу]. При цьому в динамічному представлені INDEX_STATS з'являється строчка результату аналізу. Для того щоб відстежити правильну роботу індексів адміністратор бази даних повинен за допомогою цієї команди перевірити роботу кожного індексу і прийняти рішення чи потрібно цей індекс реорганізувати чи перебудувати.

Для автоматизації роботи адміністратора бази даних пропонується метод що дозволить робити аналіз існуючих у базі даних індексів та в автоматичному режимі робити їх перебудову те реорганізацію у час, коли база даних менш всього навантажена.

Для цього використовується наступний метод.

	THEROWID	DEPARTMENT_ID	OBJID	FILENUM	BLOCKNUM	ROWSLOT
1	AAASfwAAEAAAAIVAAA	10	75760	4	533	0
2	AAASfwAAEAAAAIVAAE	12	75760	4	533	4
3	AAASfwAAEAAAAIVAAF	13	75760	4	533	5
4	AAASfwAAEAAAAIVAAG	14	75760	4	533	6
5	AAASfwAAEAAAAIVAAB	20	75760	4	533	1
6	AAASfwAAEAAAAIVAAH	23	75760	4	533	7
7	AAASfwAAEAAAAIVAAI	24	75760	4	533	8
8	AAASfwAAEAAAAIVAAC	30	75760	4	533	2
9	AAASfwAAEAAAAIVAAJ	34	75760	4	533	9
10	AAASfwAAEAAAAIVAAD	40	75760	4	533	3
11	AAASfwAAEAAAAIVAAK	43	75760	4	533	10

Рис. 3. Зміст ROWID

1. Визначаємо схеми користувачів бази даних індекси яких ми будемо обслуговувати.

2. Створюємо допоміжну таблицю в який буде зберігатись інформація про стан фрагментації індексів.

3. Запускаємо створену процедуру, яка буде по черзі вибирати зі списку схеми користувачів та в циклі по черзі аналізувати роботу кожного індексу. Після чого результати аналізу будуть переноситись з динамічного представлення INDEX_STATS в допоміжну таблицю.

4. Після завершення процедури аналізу фрагментації індексів наступна процедура буде перевіряти ступінь фрагментованності індусів і якщо вона буде більше ніж 5% та менша ніж 30% то тоді індекс буде реорганізовано якщо більше ніж 30% то індекс буде перебудовано.

На базі цього метода розроблено діаграму активностей системи, яка показана на рис. 4.

Опишемо потоків подій, які відбуваються у системі.

Варіант використання «Підключення до СУБД».

Короткий опис: дозволяє користувачу здійснити вхід до системи, використовуючи ім'я користувача та пароль.

Основний потік подій: автентифікація, починається коли користувач входить до системи.

Альтернативний потік подій: якщо користувач вводить невірні дані, з'являється відповідне повідомлення.

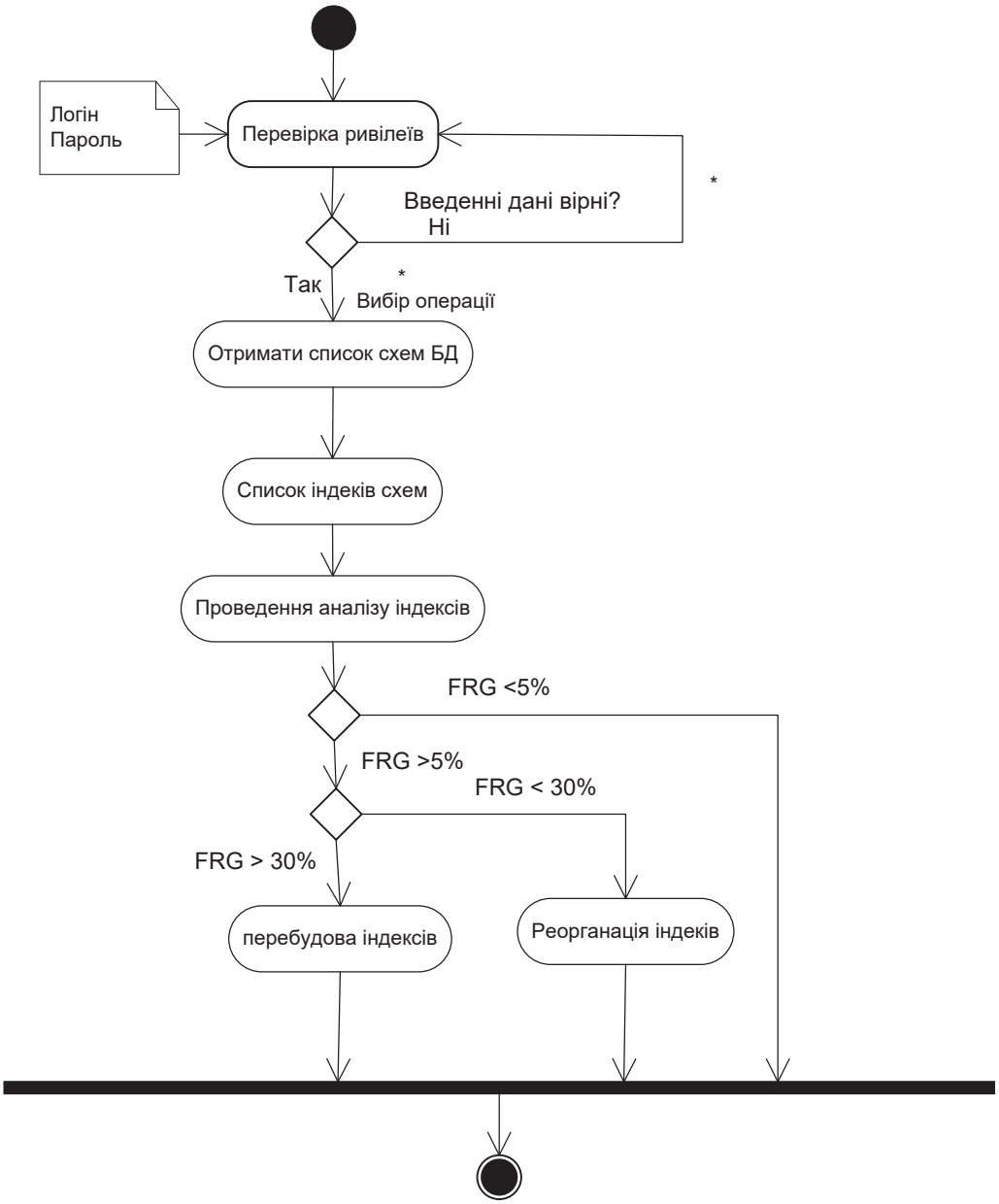

Рис. 4. Діаграма активностей системи

Передумова: запуск програми SQL+.

Постумова: Відображається вікно утиліти. SQL+ Варіант використання «Створення та запуск процедури для автоматичної перевірки фрагментації індексів БД». Короткий опис: дозволяє програмі у автоматичному режимі проводити аналіз стану індексів БД та виявити індекси, що потребують перебудови.

Основний потік подій: виконання починається, коли адміністратор бази даних запускає програму програма запускається та в автоматичному режимі формує список індексів які потрібно перебудувати.

Альтернативний потік подій: якщо відсутній зв'язок з базою даних – з'являється відповідне повідомлення.

Постумова: сформовані представлення продуктивності БД.

Варіант використання «Обробка помилок що виникли у роботі СУБД». Короткий опис: дозволяє користувачу переглядати помилки, що виникли у роботі інстанції.

Для роботи системи керування фрагментацією індексів розробляємо frontend додаток.

По-перше створюємо таблицю в якій буде зберігатись інформація стосовно фрагментації індексів.

```
SQL:> Create table INDEXFR as select *
from INDEX_STATS;
```

По-друге В базі даних може бути велика кількість користувачів но не всі користувачі мають у своїх схемах таблиці та фрагментовані індекси, які потрібно перевіряти. Тому створюємо таблицю в якій будуть зберігатись імена користувачів в схемах яких будуть зберігатись індекси, які будемо перевіряти на фрамгментацію. Адміністратор БД може самостійно добавляти чи видаляти користувачів в цю таблиці при потребі.

```
SQL:>  create table inuser (  id
number,  username nvarchar2(40))
tablespace SYSTEM
-- Create/Recreate primary, unique and
foreign key constraints
alter table inuser
  add constraint id_fr primary key (ID);
```

На прикладі СУБД Oracle створюємо процедуру яка в автоматичному режимі у зазначений адміністратором час буде перевіряти наявність фрагментованих індексів і перебудовувати їх.

```
create or replace procedure prcindfr   as
userdb varchar(40);
prcfr number;
cursor ownerfr is  select username from
INUSER;
CURSOR indstat IS
SELECT 'analyze index '||owner||'.'||index_
name||' validate structure' txtsql
     FROM DBA_INDEXES WHERE owner IN userdb;
Cursor c_reb is select i.owner, i.name,i.
del_lf_rows,i.lf_rows from INDEXFR i;
BEGIN
  execute immediate 'Delete * from
INDEXFR';
   for vownerfr in ownerfr loop
     userdb := vownerfr.username;
-- Цикл для аналізу індексів і занесення
їх до допоміжної таблиці
```

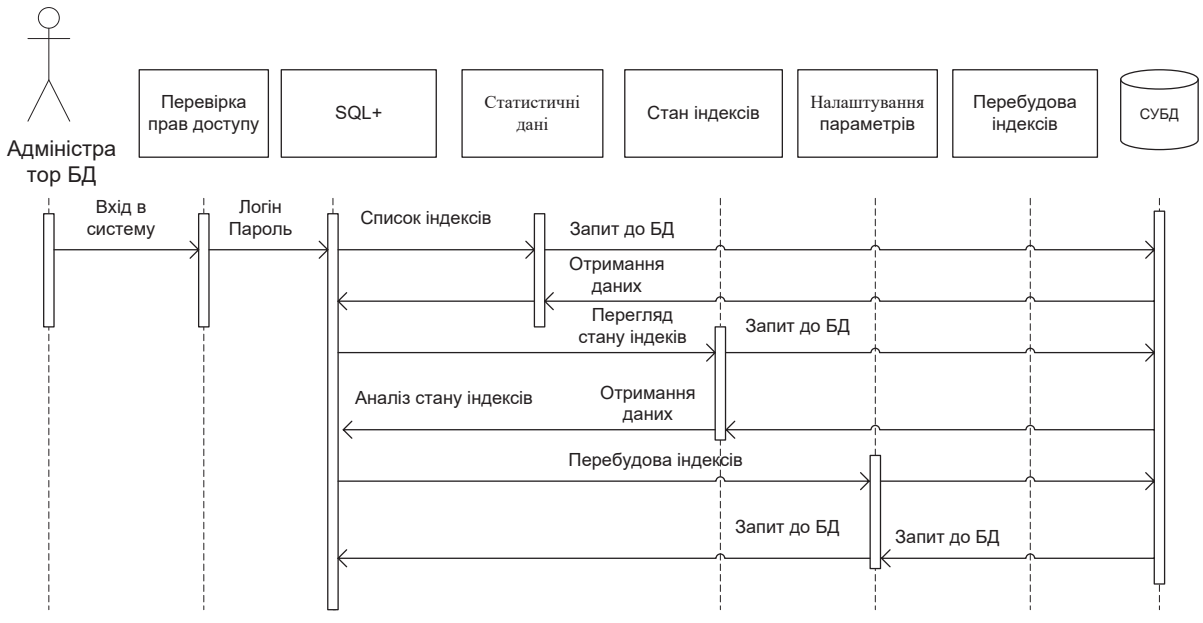

Рис. 5. Діаграма потоку подій роботи системи

```
    FOR cindstat IN indstat LOOP

           execute immediate cindstat.txtsql;
         INSERT all into INDEXFR
(name,owner,lf_rows,del_lf_rows)
         values(n,userdb ,lf,del_lf)
         SELECT name as n,lf_rows as lf,del_
lf_rows  as del_lf  FROM sys.INDEX_STATS;
         END LOOP;
         end loop;
   -- Цикл для пошуку фрагментованих індексів
та їх перебудови
         for v_reb in c_reb loop
           if v_reb.lf_rows!=0 then
         prcfr:=  v_reb.del_lf_rows/v_reb.
lf_rows;
          end if;
           if prcfr > 0.05 then
            execute immediate 'alter index'
||v_reb.owner||'.'||v_reb.name||'Rebuild';
            end if;
           end loop;
         END prcindfr;
```

Далі визначаємо час коли сервер менш всього завантажений і запускаємо цю процедуру, як SNP-процес.

```
SQL> VARIABLE v_JobNum NUMBER
SQL> BEGIN
  2      DBMS_JOB.SUBMIT(:v_JobNum,
'prcindfr;', SYSDATE,
  3    'NEXT_DAY(TRUNC(SYSDATE),
"SATURDAY")+24');
  4    END;
```

Цей скрипт дозволить автоматично запускати цю процедуру о півночі з суботи на неділю.

Висновки з даного дослідження і перспективи подальших розвідок у даному напрямку. В ході виконання роботи набув подальший розвиток метод перебудови фрагментованих індексів у базі даних. В перше запропоновано використання перебудови індексів у автоматичному режимі без втручання адміністратора бази даних. Розроблений метод дозволяє шукати фрагментовані індекси у схемах заданих користувачів в інтервали часу в які сервер менше навантажений та робити перебудову індексів. На базі розробленого методу було розроблено процедуру за допомогою якої проводиться перебудова фрагментованих індексів код якої наведено у статі.

Результати даної роботи можуть бути використані для подальших досліджень і розробок, а також для запровадження використання технології уникнення фрагментації для різних типів СУБД.

ЛІТЕРАТУРА:

1. Разрешение фрагментации индекса путем реорганизации или перестроения индекса. URL: https://docs.microsoft.com/ru-ru/sql/relational-databases/indexes/reorganize-and-rebuild-indexes?view=sql-server-ver15#.

2. Indexes and Index-Organized Tables. URL: https://docs.oracle.com/cd/E11882_01/server.112/e40540/indexiot.htm#CNCPT721.

3. Oracle Advanced Analytics Customer Success Stories. URL: https://www.oracle.com/technetwork/database/options/advanced-analytics/odm/odm-customers-086483.html.

4. Фрагментация базы данных: основные методы и случаи использования. URL: https://www.8host.com/blog/fragmentaciya-bazy-dannyx-osnovnye-metody-i-sluchai-ispolzovaniya/.

5. Steve Ataky Tsham Mpinda1, Lucas Cesar Ferreira1, Marcela Xavier Ribeiro1, Marilde Terezinha Prado Santos Evaluation of Graph Databases Performance through Indexing Techniques / International Journal of Artificial Intelligence & Applications (IJAIA) Vol. 6, No. 5, September 2015. DOI: 10.5121/ijaia.2015.6506.

6. How to Create an Index-Organized Table in Oracle 12c / Bob Bryla, Kevin Loney. URL: https://logicalread.com/index-organized-table-oracle-12c-mc06/#.YGrftugzbIU.

7. Using Index-Organized Tables in Oracle David Fitzjarrell [Electronic resource]. URL: https://www.databasejournal.com/features/oracle/using-index-organized-tables-in-oracle.html.

8. Darl Kuhn, Sam R. Alapati and Bill Padfield Oracle Indexing and Access: Apress – 2016. ISBN 978-1-4842-1984-3. DOI 10.1007/978-1-4842-1984-3.

9. Берко А.Ю., Верес О.М., Пасічник В.В., Берко А.Ю. Системи баз даних та знань: навч. посібник. Кн. 1. Організація баз даних та знань. Львів : «Магнолія 2006», 2008. 456 с. ISBN 978-966-2025-56-9.

10. Пасічник В. В., Резниченко В.А. Організація баз даних та знань / Київ : Видавнича група BHV, 2006. 384 с.

REFERENCES:

1. Razreshenie fragmentaczii indeksa putem reorganizaczii ili perestroeniya indeksa. Access mode: https://docs.microsoft.com/ru-ru/sql/relational-databases/indexes/reorganize-and-rebuild-indexes?view=sql-server-ver15#.

2. Indexes and Index-Organized Tables. Access mode: https://docs.oracle.com/cd/E11882_01/server.112/e40540/indexiot.htm#CNCPT721.

3. Oracle Advanced Analytics Customer Success Stories. Access mode: https://www.oracle.com/technetwork/database/options/advanced-analytics/odm/odm-customers-086483.html.

4. Fragmentacziya bazy` danny`kh: osnovny`e metody` i sluchai ispol`zovaniya. Access mode: https://www.8host.com/blog/fragmentaciya-bazy-dannyx-osnovnye-metody-i-sluchai-ispolzovaniya/

5. Steve Ataky Tsham Mpinda1, Lucas Cesar Ferreira1, Marcela Xavier Ribeiro1, Marilde Terezinha Prado Santos Evaluation of Graph Databases Performance through Indexing Techniques / International Journal of Artificial Intelligence & Applications (IJAIA) Vol. 6, No. 5, September 2015. DOI: 10.5121/ijaia.2015.6506

6. How to Create an Index-Organized Table in Oracle 12c / Bob Bryla, Kevin Loney. Access mode: https://logicalread.com/index-organized-table-oracle-12c-mc06/#.YGrftugzbIU.

7. Using Index-Organized Tables in Oracle David Fitzjarrell. Access mode: https://www.databasejournal.com/features/oracle/using-index-organized-tables-in-oracle.html.

8. Darl Kuhn, Sam R. Alapati and Bill Padfield Oracle Indexing and Access: Apress – 2016. ISBN 978-1-4842-1984-3. DOI 10.1007/978-1-4842-1984-3.

9. Berko, A. Yu. Sistemi baz danikh ta znan`: navch. posi`bnik. Kn. 1. Organi`zaczi`ya baz danikh ta znan` / A.Yu. Berko, O.M. Veres, V.V. Pasi`chnik. L. : “Magnoli`ya 2006”, 2008. 456 s. ISBN 978-966-2025-56-9

10. Pasi`chnik V.V., Reznichenko V.A. Organi`zaczi`ya baz danikh ta znan` / Kyiv: Vidavnicha grupa BHV, 2006. 384 p.