ວາລະສານວິທະຍາສາດມະຫາວິທະຍາໄລສຸພານຸວົງ, ຄົ້ນຄວ້າວິໄຈສະຫະສາຂາວິຊາ, ວາລະສານເປີດກວ້າງ
ສະບັບທີ 6, ເຫຼັ້ມທີ 1, ມັງກອນ-ມີຖຸນາ 2020, ເລກທະບຽນ ISSN 2521-0653

ການຈຳລອງການເພີ່ມປະສິດທິພາບການຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາມະຫາວິທະຍາໄລສຸພານຸວົງ ໂດຍການນຳໃຊ້ອິນເດັກ Bitmap ແລະ B-Tree[1]

ໄຊສະຫວາດ ສຸກຈະເລີນ, ວິນັດ ເມກທະນາວັນ[2], ເວຍລະກອນ ມະນີວັນ, ຊ້າງຢ່າງ ແລະ ທອງເພັດ ຄອງເກດ
ພາກວິຊາ ວິສະວະກຳຄອມພິວເຕີ ຄະນະວິສະວະກຳສາດ ມະຫາວິທະຍາໄລ ສຸພານຸວົງ

ບົດຄັດຫຍໍ້

ປັດຈຸບັນໄດ້ມີການນຳຂໍ້ມູນຫຼາຍປະເພດເຂົ້າມານຳໃຊ້ໃນລະບົບຂໍ້ມູນຂ່າວສານ ດັ່ງນັ້ນຖານຂໍ້ມູນຖືວ່າເປັນສິ່ງສຳຄັນໃນການເກັບກຳຂໍ້ມູນຂອງອົງກອນ ເຊິ່ງປຽບເໝືອນຫົວໃຈຫຼັກຂອງອົງກອນ ແລະ ມີຄວາມສຳຄັນໃນການນຳໃຊ້ເຂົ້າໃນວຽກງານດ້ານຕ່າງໆ ໂດຍສະເພາະທາງດ້ານເຕັກນິກ ການໃຊ້ຄຳສັ່ງ ໃນການຄົ້ນຫາຂໍ້ມູນຈາກຖານຂໍ້ມູນນັ້ນ ແລະ ມີຫຼາຍຮູບແບບ ແຕ່ອາດຈະໃຊ້ເວລາໃນການປະມວນຜົນແຕກຕ່າງກັນ.ໃນບົດວິໄຈນີ້ ພວກເຮົາໄດ້ເພີ່ມປະສິດທິພາບການຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາຂອງມະຫາວິທະຍາໄລສຸພານຸວົງ ໂດຍການນຳໃຊ້ ອິນເດັກ Bitmap ແລະ B-tree ຫຼັງຈາກນັ້ນໄດ້ສົມທຽບລະຫວ່າງການຄົ້ນຫາຂໍ້ມູນທີ່ບໍ່ໄດ້ມີການນຳໃຊ້ອິນເດັກ. ການທົດລອງໃນການຄົ້ນຫາຂໍ້ມູນນັກສຶກສາໂດຍການນຳໃຊ້ ອິນເດັກ Bitmap ແລະ B-treeໃນລະບົບຖານຂໍ້ມູນນັ້ນປະກອບມີເຊັ່ນ: ຊີວະປະຫວັດ, ຄະແນນ, ການລົງທະບຽນ, ແລະ ລາຍວິຊາຮຽນ ຜົນຂອງການທົດລອງສະແດງໃຫ້ເຫັນວ່າ ການຄົ້ນຫາຂໍ້ມູນນັກສຶກສາໂດຍການນຳໃຊ້ ອິນເດັກ ທັງສອງຮູບແບບນັ້ນແມ່ນມີປະສິດທິພາບດີກວ່າ ການຄົ້ນຫາແບບທຳມະດາ ເຊິ່ງສະແດງອອກເວລາທີ່ໃຊ້ໃນການຄົ້ນຫາຂອງການທົດລອງ.

ຄຳສຳຄັນ: ການເພີ່ມປະສິດທິພາບ, ອິນເດັກ Bitmap ແລະ ອິນເດັກ B-tree

[1] ***ການອ້າງອີງພາສາລາວ:*** ໄຊສະຫວາດ ສຸກຈະເລີນ ແລະ ຄະນະ. (2020). **ການຈຳລອງການເພີ່ມປະສິດທິພາບການຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາມະຫາວິທະຍາໄລສຸພານຸວົງໂດຍການນຳໃຊ້ ອິນເດັກ Bitmap ແລະ B-Tree**, *ວາລະສານວິທະຍາສາດ ມະຫາວິທະຍາໄລ ສຸພານຸວົງ, ເລກທະບຽນ ISSN 2521-0653, ສະບັບທີ: 6, ເຫຼັ້ມທີ 1, ໜ້າທີ: 196 - 205.*

[2] ***ຊື່ຜູ້ຕິດຕໍ່ພົວພັນ:*** *ວິນັດ ເມກທະນາວັນ, ພາກວິຊາ ວິສະວະກຳຄອມພິວເຕີ,ຄະນະວິສະວະກຳສາດ,ມະຫາວິທະຍາໄລ ສຸພານຸວົງ ໂທ: 020 5540 7070, E-mail: vinath.mek@gmail.com*

Modelling optimize of searching student's information in Souphanouvong University by using Bitmap and B-tree index methods

Xaysavath SOUKCHALEUN, Vinath MEKTHANAVANH[3], Vialakhone MANIVANH, Xangyang[9] and Thongphet KHONGKET

Department of Computer Engineering, Faculty of Engineering, Souphanouvong University

Abstract

Currently, there are many types of data used in the database system. Database is an important for the organization's data collection. It is a centralization of the organization for using in various work by implementing different techniques to search information from database. Different techniques, the time for processing is different. In this paper, we increased the effectiveness of searching student information of Souphanouvong University by using Bitmap index and B-tree index. "After that" we compared the results between the technique of with and without using indexes. In the experiment, we used three attributes of student information there are scores, registration, and subjects. Experimental results show that the time using in both types of indexes outperformed simple technique.

.

Keywords: Optimization, Bitmap index, B-tree index

[3] ***Corresponding Author:*** *Vinath MEKTHANAVANH, Department of Computer Engineering, Faculty of Engineering, Souphanouvong University), Tel: 020 5540 7070 E-mail: vinath.mek@gmail.com*

1. ພາກສະເໜີ

ປັດຈຸບັນພວກເຮົາໄດ້ມີການນຳໃຊ້ຖານຂໍ້ມູນເຂົ້າໃນວຽກງານຕ່າງໆ ເຊັ່ນ: ສະຖາບັນການສຶກສາ, ທຸລະກິດ, ທະນາຄານ, ໂຮງໝໍ, ຮ້ານຄ້າ ແລະ ອື່ນໆ ອີກຫຼາຍຢ່າງ ເຊິ່ງການບັນທຶກຂໍ້ມູນເຫຼົ່ານີ້ລ້ວນແຕ່ມີຂໍ້ມູນຂະໜາດໃຫຍ່ເຊິ່ງສິ່ງຜົນສະທ້ອນໃຫ້ຕ່ຳການຄົ້ນຫາຂໍ້ມູນໃນການສະແດງຜົນໃນຮູບແບບໜ້າເວບ ແລະ ລາຍງານຕ່າງໆມີຄວາມຊັກຊ້າ ໃນບາງຄັ້ງລະບົບຖານຂໍ້ມູນບໍ່ສາມາດຕອບສະໜອງການເຮັດວຽກຂອງຜູ້ໃຊ້ໄດ້ດັ່ງນັ້ນຈຶ່ງຮຽກຮ້ອງໃຫ້ມີການເພີ່ມປະສິດທິພາບໃນລະບົບຈັດການຖານຂໍ້ມູນເພື່ອໃຫ້ສາມາດປະຍັດເວລາໃນການຄົ້ນຫາຂໍ້ມູນໄດ້ໄວຂຶ້ນ [1] .

ລະບົບຖານຂໍ້ມູນ ເປັນແຫຼ່ງເກັບຂໍ້ມູນທີ່ມີຂະໜາດໃຫ່ຍ ການຄົ້ນຂໍ້ມູນທີ່ມີລັກສະນະການປະມວນຜົນແບບອອນລາຍ Online Analytical Processing (OLAP) ເປັນການຄົ້ນຫາຂໍ້ມູນເພື່ອຊ່ວຍໃນການຕັດສິນໃຈຂອງຜູ້ບໍລິການ [2] . ເຊິ່ງມີຄວາມຊັບຊ້ອນ ແລະ ເປັນການຕອບໂຕ້ແບບທັນທີທັນໃດໂດຍ ຂໍ້ມູນທີ່ເກັບໃນແຫຼ່ງເກັບຂໍ້ມູນນັ້ນແມ່ນຂໍ້ມູນທີ່ມາຈາກຫຼາຍຕາຕະລາງ ແລະ ຂໍ້ມູນທີ່ເກັບຕັ້ງແຕ່ອາດີດຈົນເຖິງປັດຈຸບັນ ຂໍ້ມູນທີ່ຢູ່ໃນຄັງຂໍ້ມູນຈຶ່ງມີປະລິມານຫຼາຍເຮັດໃຫ້ການປະມວນຜົນໃນການຄົ້ນຫາຂໍ້ມູນຕ້ອງໃຊ້ເວລາຫຼາຍຂຶ້ນ [3] .

ການເພີ່ມປະສິດທິພາບໃນການຄົ້ນຫາຂໍ້ມູນມີຫຼາຍວິທີ ແຕ່ລະວິທີກໍ່ມີຄວາມ ແຕກຕ່າງກັນເຊັ່ນ: ການປະມວນຜົນແບບຄູ່ຂະໜານ ແລະ ການສ້າງ ອິນເດັກ ເຊິ່ງການສ້າງ ອິນເດັກ ເປັນເຕັກນິກ.ການສ້າງເພີ່ມຄວາມໄວທີ່ບໍ່ຕ້ອງເສຍຄ່າໃຊ້ຈ່າຍໃນການເພີ່ມຮາດແວ ສຳລັບລະບົບຄັງຂໍ້ມູນທີ່ນຳໃຊ້ຄື ອິນເດັກ Bitmap ແລະ B-Tree[4].ເນື່ອງຈາກສາມາດດຳເນີນໃນລະດັບບິດ (Bit) ແລະ ໄບນາລີ (Binary) ນອກຈາກນີ້ການສ້າງ Bitmap ແລະ B-Tree ຍັງມີປະສິດທິພາບໃນການໃຊ້ເນື້ອທີ່ໃນການຈັດເກັບ ອິນເດັກ ໂດຍສະເພາະໃນ ເອດທຣິບິວ (attribute) ທີ່ມີຄ່າ Cardinalin ຕ່ຳ (Cardinality ຄືຈຳນວນຄ່າທີ່ແຕກຕ່າງກັນໃນຖັນ Column ທີ່ນຳມາສ້າງ ອິນເດັກ [5].

ການຄົ້ນຫາຂໍ້ມູນນັກສຶກສາຂອງມະຫາວິທະຍາໄລ ສຸພານຸວົງ ປະຈຸບັນ ຍັງບໍ່ໄດ້ສ້າງ ອິນເດັກ ທີ່ເໝາະສົມກັບການຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາ. ສະນັ້ນຈຶ່ງເຮັດໃຫ້ການ ຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາມີຄວາມຫຍຸ້ງຍາກ ແລະ ຊັກຊ້າ, ບໍ່ສາມາດຕອບສະໜອງກັບຄວາມຕ້ອງການຂອງຜູ້ໃຊ້ເທົ່າທີ່ຄວນໄດ້.

ຈາກບັນຫາດັ່ງກ່າວຈຶ່ງເປັນແຮງຈູງໃຈໃຫ້ພວກເຮົາມີການວິໄຈເພື່ອເພີ່ມປະສິດທິພາບໃຫ້ແກ່ການຈຳລອງລະບົບການຈັດການຖານຂໍ້ມູນໃນລະບົບການຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາມະຫາວິທະຍາໄລສຸພານຸວົງ ດ້ວຍວິທີການສ້າງ ອິນເດັກ ໃນຮູບແບບ Bitmap ແລະ B-Tree [6] .

1. ນິຍາມ ແລະ ວິທີການ

ອິນເດັກ ມີຄວາມສຳຄັນໃນການເພີ່ມຄວາມໄວ ແລະ ປະສິດທິພາບໃນການເຂົ້າເຖິງຂໍ້ມູນ ເຊິ່ງຊ່ວຍຫຼຸດຈຳນວນການເຂົ້າເຖິງຂໍ້ມູນ ສິ່ງຜົນເຮັດໃຫ້ຈຳນວນການທຳງານຂອງ input ແລະ output ຕ່ຳການຄົ້ນຫາຂໍ້ມູນຫຼຸດລົງ. ການນຳໃຊ້ອິນເດັກ ແມ່ນຈະຕ້ອງໄດ້ສ້າງອິນເດັກຂຶ້ນມາກ່ອນ ເຊິ່ງການສ້າງອິນເດັກນັ້ນ ອາດຈະສ້າງຢູ່ຖານຂໍ້ມູນ ຫຼື ຜູ້ໃຊ້ສ້າງຂຶ້ນມາເອງໂດຍທີ່ ກໍລະນີທີ່ສ້າງ ຫຼື ລຶບ ອິນເດັກ ອອກຈະບໍ່ມີຜົນກະທົບຕໍ່ຂໍ້ມູນໃນຖານຂໍ້ມູນ ແຕ່ປະສິດທິ ພາບໃນການຄົ້ນຫາຂໍ້ມູນອາດຈະຫຼຸດລົງ. ອິນເດັກເປັນຕົວຊ່ວຍໃນການເຂົ້າຫາຂໍ້ມູນເຊິ່ງມີໜ້າທີ່ຊີ້ບອກຕຳແໜ່ງທີ່ຢູ່ ຂອງຂໍ້ມູນນັ້ນໂດຍກົງພາຍຫຼັງທີ່ມີການສ້າງອິນເດັກແລ້ວ ລະບົບຈັດການຖານຂໍ້ມູນຈະມີໜ້າທີ່ໃນການດູແລ ແລະ ນຳໃຊ້ອິນເດັກນັ້ນຕໍ່ໄປອິນເດັກທີ່ສ້າງຂຶ້ນຈະເກີດມີການປ່ຽນແປງເມື່ອມີການເພີ່ມແກ້ໄຂ ຫຼື ລຶບໃນຖານຂໍ້ມູນ ທີ່ກ່ຽວຂ້ອງກັບ ອິນເດັກທີ່ສ້າງຂຶ້ນມານັ້ນ.ຈາກຂໍ້ມູນດັ່ງກ່າວເຮົາສາມາດໃຊ້ ອິນເດັກໄດ້ຫຼາຍກວ່າໜຶ່ງອິນເດັກໃນໜຶ່ງຕາຕະລາງຂອງຖານຂໍ້ມູນໄດ້ ແຕ່ການສ້າງ ອິນເດັກ ເປັນການເພີ່ມກິດຈະກຳ ໃຫ້ກັບລະບົບ ເຊິ່ງຖານຂໍ້ມູນທີ່ຈະຕ້ອງດູ

ແລ ອິນເດັກ ຕົວ ນັ້ນຕໍ່ໄປ ແລະ ເປັນການເພີ່ມການໃຊ້ເນື້ອທີ່ ສຳລັບຈັດເກັບ ອິນເດັກທີ່ສ້າງຂຶ້ນ. ການເກັບຄ່າຂອງອິນເດັກ ສາມາດແບ່ງອອກເປັນ 2 ປະເພດຄື [7] [8] [9] [10] :

- ການເກັບຄ່າອິນເດັກທີ່ບໍ່ຊ້ຳກັນ ໂດຍ ໃຊ້ກັບຄີຫຼັກຂອງຕາຕະລາງໃນຖານຂໍ້ມູນ
- ການເກັບຄ່າອິນເດັກທີ່ຊ້ຳກັນໄດ້

ຕົວຢ່າງ: ການໃຊ້ ອິນເດັກກັບລະບົບວຽກຊີວິດປະຈຳວັນ.

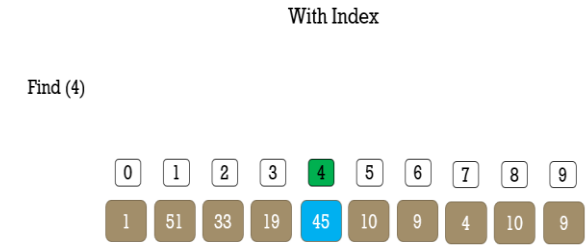
ສົມມຸດວ່າຕູ້ເອກະສານຕູ້ໜຶ່ງມີ 15 ຊັ້ນ ສຳລັບເກັບຮັກສາແຟ້ມເອກະສານຂໍ້ມູນນັກສຶກສາໃນຄະນະວິຊາໃດໜຶ່ງໂດຍບໍ່ມີການຈັດລຽງເອກະສານໃດໆ ຖ້າເຮົາຕ້ອງການຄົ້ນຫາຂໍ້ມູນສຳລັບນັກສຶກສາທີ່ມີຊື່ທ້າວ ກ ວິທີທີ່ເຮົາຕ້ອງໃຊ້ຄື ເປີດທຸກຊັ້ນ ແລະ ເບິ່ງທຸກແຟ້ມໃນແຕ່ລະຊັ້ນຊັ້ນເພື່ອຊອກຫາຂໍ້ມູນນັກສຶກສາຄົນນັ້ນ ເຊິ່ງໃນກໍລະນີນີ້ເກີດມີຄວາມຊັກຊ້າໂດຍຈະເລີ່ມຈາກການຄົ້ນຫາຈາກຊັ້ນເທິງສຸດຫາຊັ້ນລຸ່ມສຸດຈຶ່ງຈະພົບ ແຕ່ວິທີການແກ້ໄຂກໍ່ຄືໃຊ້ວິທີການຈັດລຽງຕົວອັກສອນ ແລະ ເກັບຮັກສາແຟ້ມຂໍ້ມູນຕາມຕົວອັກສອນ. ເມື່ອປຽບທຽບກັບອິນເດັກຢູ່ໃນລະບົບຖານຂໍ້ມູນກໍ່ຄື ອິນເດັກຈະເປັນໂຕຊີ້ບອກທີ່ຢູ່ຂອງຕົວອັກ

ສອນ ເພື່ອລະບຸຫາຂໍ້ມູນທີ່ເຮົາຕ້ອງການຄົ້ນຫາ [11] .

ການອອກແບບຖານຂໍ້ມູນ ເປັນການອອກແບບເພື່ອຈັດເກັບຂໍ້ມູນ ແລະ ເອີ້ນໃຊ້ຂໍ້ມູນຢ່າງມີປະສິດທິພາບ. ອິນເດັກເປັນວິທີການໜຶ່ງທີ່ນຳມາໃຊ້ເພື່ອເພີ່ມປະສິດທິພາບໃນການຄົ້ນຫາຂໍ້ມູນເຊິ່ງການສ້າງ ອິນເດັກ ສາມາດເຮັດໄດ້ໃນລະບົບຈັດການຖານຂໍ້ມູນໂດຍການໃຊ້ຄຳສັ່ງ SQL ເຂົ້າໃນການສ້າງ ອິນເດັກ. ດັ່ງນັ້ນໃນການອອກແບບຖານຂໍ້ມູນ ຜູ້ອອກແບບນັ້ນຕ້ອງ

ຕັດສິນໃຈວ່າຂໍ້ມູນທີ່ຖືກອອກແບບຈະມີໂຄງສ້າງການຈັດເກັບແນວໃດໃນໜ່ວຍຄວາມຈຳສຳຮອງ ເພື່ອເຮັດໃຫ້ການເຮັດວຽກຢ່າງມີປະສິດທິພາບ [12] .

ຖ້າປຽບທຽບການໃຊ້ງານອິນເດັກໃນ MySQL ກໍ່ຄືກັບອິນເດັກຢູ່ໃນຕອນທ້າຍຂອງປຶ້ມຕຳລາ ຕາມປົກກະຕິແລ້ວຖ້າເຮົາຕ້ອງການຄົ້ນຫາເນື້ອໃນໃດທີ່ສົນໃຈເປັນພິເສດໃນຕຳລາກໍ່ຈະເປີດເບິ່ງໜ້າອິນເດັກກ່ອນ ເຊິ່ງໜ້ານີ້ຈະມີຂໍ້ມູນບອກໃຫ້ຮູ້ວ່າ ເນື້ອໃນທີ່ເຮົາສົນໃຈຢູ່ນັ້ນແມ່ນທີ່ໜ້າໃດ ເຮັດໃຫ້ເຮົາສາມາດໄປເປີດເບິ່ງຂໍ້ມູນຢູ່ໜ້ານັ້ນໄດ້ທັນທີໂດຍບໍ່ຕ້ອງມາເປີດຄົ້ນຫາທີລະໜ້າ [13] .



ຮູບທີ 1. ການຄົ້ນຫາແບບອິນເດັກ

ຈາກຮູບທີ 1 ເຮົາ ຊອກຫາຄີທີ່ 45 ເຮົາບໍ່ຈຳເປັນຕ້ອງຊອກຫາເທື່ອລະຄີ ເພາະເຮົາໄດ້ສ້າງອິນເດັກຢູ່ໃນລະບົບຖານຂໍ້ມູນນັ້ນກໍ່ຄື ອິນເດັກຈະເປັນໂຕຊີ້ບອກທີ່ຢູ່ຂອງຄີທີ່ 45 ທີ່ລະບຸໄວ້.

ອິນເດັກ ທີ່ໃຊ້ໃນ MySQL ກໍ່ມີລັກສະນະການເຮັດວຽກຄືກັນກັບ MySQL ໂດຍຈະເກັບຂໍ້ມູນ ອິນເດັກ ຈັດລຽງຕາມລຳດັບໄວ້ເພື່ອຊ່ວຍໃນການຄົ້ນຫາ ເຊິ່ງ ຂໍ້ມູນທີ່ເກັບຢູ່ໃນອິນເດັກຈະເຊື່ອມໂຍງໄປຫາຂໍ້ມູນຈິງໃນຖານຂໍ້ມູນອີກເທື່ອໜຶ່ງ.

ເພື່ອໃຫ້ເຫັນພາບການເຮັດວຽກຂອງ ອິນເດັກ ໃນ MySQL ພ ວ ເ ຮົ າ ໄ ດ້ ສ້ າ ງ ຕ າ ຕ ະ ລ າ ງ course_student ເພື່ອໃຫ້ເຫັນການເຮັດວຽກຂອງ ອິນເດັກ ຕາຕະລາງນີ້ມີ course_id ເກັບຂໍ້ມູນຂອງ ລະຫັດວິຊາ ກັບຖັນ

ນັກສຶກສາທີ່ລົງທະບຽນວິຊາດັ່ງລຸ່ມນີ້ [14] :

ຕາຕະລາງທີ 1. ການຄົ້ນຫາລະຫັດ

course_id	student_id
1001	4701
1001	4702
1001	4703
1002	4705
1003	4702
1003	4703
1004	4701
1004	4703
1005	4704
1006	4704
1007	4701
1007	4704
1008	4705
1008	4704

ສົມມຸດເຮົາຕ້ອງການຮູ້ວ່າ ນັກສຶກສາທີ່ມີລະຫັດ 4701 ໄດ້ລົງທະບຽນວິຊາໃດແດ່ ຖ້າ MySQL ຈະຄົ້ນຫາ ຂໍ້ມູນໃນ Student_id ຕັ້ງແຕ່ເຣັກຄອດທຳອິດຈົນໄປຮອດເຣັກຄອດສຸດທ້າຍ ເພື່ອຊອກຫາເຣັກຄອດທີ່ມີລະຫັດນັກສຶກສາເປັນ 4701 ເຊິ່ງເປັນວິທີການທີ່ບໍ່ມີປະສິດຕິພາບ ໂດຍສະເພາະຢ່າງຍິ່ງໃນກໍລະນີທີ່ຕາຕະລາງມີຂໍ້ມູນເກັບຢູ່ຈຳນວນຫຼາຍ ເຮັດໃຫ້ເສຍເວລາຫຼາຍ ເພາະອາດຈະມີຂໍ້ມູນທີ່ຕ້ອງການພຽງເຣັກຄອດດຽວ ແຕ່ຕ້ອງກວດສອບຂໍ້ມູນທຸກເຣັກຄອດ.ຕາຕະລາງຕໍ່ໄປຄືເປັນຕາຕະລາງທີ່ມີການສ້າງ ອິນເດັກ ສຳລັບຖັນ student_id ດັ່ງຕາຕະລາງລຸ່ມນີ້:

ຕາຕະລາງທີ 2. ການຄົ້ນຫາລະຫັດ

course_id	student_id		Index
1001	4701		4701
1001	4702		4701
1001	4703		4701
1002	4705		4702
1003	4702		4702
1003	4703		4703
1004	4701		4703
1004	4703		4703
1005	4704		4704
1006	4704		4704
1007	4701		4704
1007	4704		4704
1008	4705		4705
1008	4704		4705

ຈາກຕາຕະລາງທີ 2 ສະແດງໃຫ້ເຫັນວ່າເມື່ອມີການນຳໃຊ້ ອິນເດັກ ມາຊ່ວຍລະບົບຈະສາມາດຄົ້ນຫາຂໍ້ມູນທີ່ຕ້ອງການໄດ້ໄວເຊິ່ງຕ້ອງການກວດສອບວ່ານັກສຶກສາທີ່ມີລະຫັດ 4701 ໄດ້ລົງທະບຽນຮຽນວິຊາໃດແດ່ ແລະ ການ ຄົ້ນຫາຂໍ້ມູນໃນ ອິນເດັກ ທີ່ມີຄ່າ 4701 ໂດຍມີລິ້ງໄປຍັງຂໍ້ມູນຈິງໄດ້ທັນທີ ແລະ ເມື່ອສຳເລັດການອ່ານຂໍ້ມູນ 4701 ແລ້ວ ແລະ ການຄົ້ນຫາກໍຈະຢຸດໂດຍທັນທີ.

1.1 ອິນເດັກ B-Tree

ເປັນອິນເດັກຫຼັກຂອງຖານຂໍ້ມູນ ເຊິ່ງມີປະສິດທິພາບສູງໃນການເອີ້ນໃຊ້ຄີຫຼັກຂອງຕາຕະລາງ ຫຼື ເອີ້ນຂໍ້ມູນທີ່ມີຜົນຮັບຈຳນວນໜ້ອຍອອກມາ ເຊິ່ງປະກອບມີ 3 ປະເພດດັ່ງນີ້ [15] :

2.1.1 ຕາຕະລາງການຈັດອິນເດັກ ຫຼື ການຈັດກຸ່ມອິນເດັກ

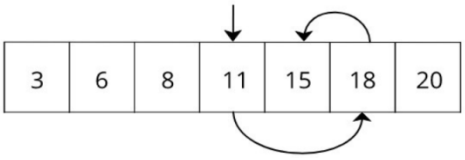
ເປັນອິນເດັກເມື່ອເວລາບັນທຶກຂໍ້ມູນໃໝ່ເຂົ້າໃນຕາຕະລາງ ຂໍ້ມູນຈະຖືກນຳໄປລຽງໄວ້ຕາມຄີຂອງອິນເດັກທີ່ໄດ້ປະກາດໄວ້ເຊັ່ນ: ເຮົາຕ້ອງເພີ່ມຂໍ້ມູນນັກສຶກສາໃໝ່ເຂົ້າໃນຖານຂໍ້ມູນ ໃນກໍລະນີລະຫັດນັກສຶກສາຖືກໃຊ້ເປັນອິນເດັກລະຫັດນັກສຶກສາຄົນກ່ອນໜ້າຄົນນີ້ມີລະຫັດນັກສຶກສາເປັນ 1001 ແລະ ນັກສຶກສາທີ່ເພີ່ມເຂົ້າມາໃໝ່ມີລະຫັດເປັນ 1002 ດັ່ງນັ້ນ: ເຣັກຄອດ ຂອງນັກສຶກສາໃໝ່ຈະຖືກເກັບຖັດຈາກເຣັກຄອດ ນັກສຶກສາກ່ອນໜ້າຢູ່ສະເໝີ ໂດຍຮູບແບບການຈັດລຽງນີ້ເອີ້ນວ່າຮູບແບບ B-Tree.

2.1.2 ການຈັດອິນເດັກສຳຮອງ ຫຼື ອິນເດັກທີ່ບໍ່ມີການຈັດກຸ່ມ.

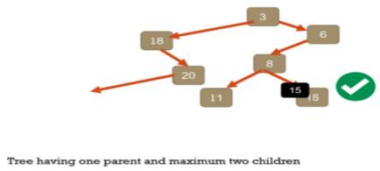
ເປັນອິນເດັກທີ່ໃຊ້ຕິອຊີ້ບອກຢູ່ໃນຕາຕະລາງການຈັດອິນເດັກ, ມີໂຄງສ້າງແບບອິດສະຫຼະ ແລະ ອິນເດັກຈະບໍ່ໄດ້ເກັບໄວ້ໃນຕາຕະລາງການຈັດອິນເດັກ ໂດຍອາໃສຕິອຊີ້ບອກເອີ້ນວ່າ ເລກແຖວ (logical rowids) ເຊິ່ງຂໍ້ມູນຈະເປັນຄ່າຂອງອິນເດັກ ເພື່ອຊີ້ໄປຍັງຕຳແໜ່ງຂອງອິນເດັກແບບຈັດລຽງຂໍ້ມູນ ສຳລັບອິນເດັກປະເພດນີ້ຈະມີປະສິດທິພາບໃນການຄົ້ນຫາຂໍ້ມູນທີ່ບໍ່ໄດ້ໃຊ້ຄີຫຼັກຂອງຕາຕະລາງ.

2.1.3 ການຈັດກຸ່ມອິນເດັກ B-Tree

ຕາຕະລາງການຈັດກຸ່ມທີ່ໃຊ້ອິນເດັກ B-Tree ໃນການບອກຕຳແໜ່ງຂອງຂໍ້ມູນດ້ວຍ ຄີຂອງການຈັດກຸ່ມ, ອິນເດັກປະເພດນີ້ຕ້ອງໄດ້ບັນທຶກຂໍ້ມູນໃສ່ໃນແຖວທຳອິດຂອງຕາຕະລາງໃນຖານຂໍ້ມູນ ການຈັດກຸ່ມຕາຕະລາງໝາຍເຖິງການຈັດຕາຕະລາງຫຼາຍຕາຕະລາງຮ່ວມກັນໃນບ້ອກດຽວກັນ ໂດຍທີ່ຂໍ້ມູນທີ່ມີຄີການຈັດກຸ່ມດຽວກັນຈະຖືກຮັກສາໄວ້ໃນບ້ອກດຽວກັນ ການຄົ້ນຫາຂໍ້ມູນໃນຮູບແບບລັກສະນະນີ້ແມ່ນມີການໃຊ້ຫຼາຍຕາຕະລາງຮ່ວມກັນຢູ່ສະເໝີ ເຊິ່ງເຮັດໃຫ້ການຄົ້ນຫາມີປະສິດທິພາບ.



ຮູບທີ 2 . ຮູບແບບການຄົ້ນຫາອິນເດັກ B-Tree



ຮູບທີ 3 . ສະແດງການຄົ້ນຫາອິນເດັກ B-Tree

ໃນທີ່ນີ້ພວກເຮົາຕ້ອງການຊອກຫາຄີທີ່ 15 ຈາກອາເລ 3,6,8,11,15 ແລະ 18 ທີ່ຖືກຈັດລຽງລຳດັບ. ຖ້າທຳການຄົ້ນຫາແບບທຳມະດາ,ມັນຈະໃຊ້ເວລາເພື່ອຄົ້ນຫາເທື່ອລະຄີ. ແຕ່ໃນການຄົ້ນຫາແບບ B-tree ຈະໃຊ້ການຄົ້ນຫາຈະຖືກພຽງແຕ່ສາມຂັ້ນຕອນ ຄື:ໂດຍທີ່ຈະເລີ່ມຈາກຕົວເລກທີ່ 11 ແລ້ວ ໄປທາງເບື້ອງຂວາ ເລກທີ່ 18 ຫຼັງຈາກນັ້ນຈະກັບມາທາງເບື້ອງຊ້າຍ.ໂດຍຫລັກການຂອງອິນເດັກ B-tree ແລ້ວຕົວເລກທີ່ໃຫ່ຍກວ່າໄປທາງເບື້ອງຂວາ ແລະ ຕົວເລກທີ່ນ້ອຍກວ່າໄປທາງເບື້ອງຊ້າຍຢູ່ສະເໝີດັ່ງທີ່ຮູບທີ 2.

$$h_{min}=[\log(n+1)]-1 \qquad (1)$$

ໃຫ້ $h \geq 1$ ເປັນຄ່າຄວາມສູງຂອງ B-tree ແລະ ໃຫ້ $n \geq 0$ ເປັນຕົວເລກທີ່ນຳເຂົ້າໃນ B-tree ໃຫ້ເປັນ m ເຊິ່ງເປັນຄ່ານ້ອຍທີ່ສຸດຂອງໂນດລູກທີ່ສາມາດມີໄດ້ 1 ໂນດແຕ່ລະໂນດແຕ່ລະໂນດສາມາດມີຫຼາຍທີ່ສຸດ m-1 ຄີ.

1.2 ອິນເດັກ Bitmap

ເປັນອິນເດັກທີ່ມີຄີຊີ້ບອກຕຳແໜ່ງຂໍ້ມູນໄດ້ຫຼາຍແຖວ ເຊິ່ງຕ່າງຈາກ B-Tree ທີ່ສາມາດຊີ້ຕຳແໜ່ງໄດ້ເທື່ອລະແຖວເທົ່ານັ້ນ: ເຜດນັກສຶກສາ, ໂດຍ Bitmap ຈະສ້າງຕາຕາລາງສຳລັບເກັບຄ່າແຕ່ລະແຖວ ໂດຍມີຂະໜາດເທົ່າກັບຈຳນວນແຖວຂອງຂໍ້ມູນຄູນກັບຈຳນວນຄ່າທີ່ແຕກຕ່າງກັນຂອງຄີ ແລະ ສາມາດຊີ້ໄປຍັງແຖວຂອງຂໍ້ມູນຈິງໄດ້ [18] .

$$\text{Bitmap} = nL \qquad (2)$$

Bitmap ແມ່ນຕາຕະລາງ ອິນເດັກ Bitmap ສຳລັບເກັບຄ່າແຕ່ລະແຖວຂອງນັກສຶກສາ, n ແມ່ນຈຳນວນແຖວ ແລະ L ແມ່ນຈຳນວນຖັນ.

ຕາຕະລາງທີ 3. ອິນເດັກ Bitmap

Table Data				Bitnam index					
ID	Gender	Married	Age range	Female	Married	18-34	34-49	50-62	62+
1001	F	N	18-34	1	0	1	0	0	0
1002	F	Y	35-49	1	1	1	0	0	0
1003	M	Y	35-49	0	1	0	1	0	0
1004	F	Y	35-49	1	1	0	1	0	0
1005	F	Y	18-49	1	1	1	0	0	0
1006	M	Y	18-34	0	1	0	0	0	1
1007	M	N	62+	1	0	0	0	1	0
1008	F	Y	50-61	0	1	0	1	0	0
1009	M	Y	35-49	1	1	0	1	0	0
1010	M	N	35-49	0	0	0	1	0	0
1011	M	Y	35-49	0	1	0	1	0	0
1012	F	N	18-34	1	0	1	0	0	0

ໃນກໍລະນີຊອກຫາຂໍ້ມູນນັກສຶກສາຄື: ເຜດ, ສະຖະນາແຕ່ງງານ ແລະ ອາຍຸຢູ່ລະຫວ່າງ 35-49 ປີວ່າມີນັກສຶກສາຈັກຄົນ ແລະ ນັກສຶກສາຄົນໃດແທ້. ຈາກຕາຕະລາງທີ 3 ພົບວ່າມີນັກສຶກສາ 2 ຄົນຄື: ລະຫັດນັກສຶກສາ 1004 ແລະ 1009. Bitmap Join ເປັນ Bitmap ອິນເດັກ ທີ່ໃຊ້ການເຮັດວຽກຮ່ວມກັບຫຼາຍຕາຕະລາງ ຂໍ້ມູນໂດຍສ້າງຕາຕະລາງທີ່ເກັບຄ່າເລກແຖວກັບຄ່າທີ່ຕ້ອງການຂອງອີກຕາຕາລາງ ຄືເຮົາເກັບຄ່າຈາກອີກຕາຕະລາງທີ່ເຮົາຕ້ອງການເຮັດວຽກໄວ້ກ່ອນເພື່ອຫຼຸດການດຶງຂໍ້ມູນຈາກອີກຕາຕາລາງ.

ตาตะลางที 4. ອິນເດັກ Bitmap

RID	...	X	...	E_2	E_1	E_0
1		B		0	0	1
2		C		0	1	0
3		A		0	0	0
4		H		1	1	1
5		A		0	0	0
6		G		1	1	0
7		D		0	1	1
8		D		0	1	1
9		F		1	0	1
.		.		.	.	.
.		.		.	.	.
.		.		.	.	.
10000		E		1	0	0

Mapping table

A	000
B	001
C	010
D	011
E	100
F	101
G	110
H	111

1.3 ວິທີການສ້າງ ອິນເດັກ B-tree

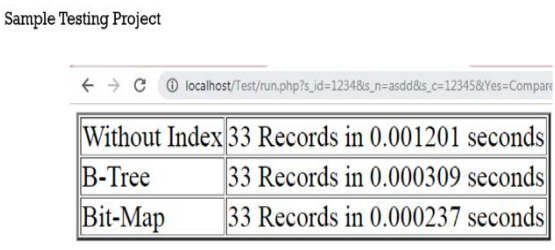
ຕົວຢ່າງ: ຄໍ່ສັ່ງ SQL ແບບ B-tree ທີ່ຢູ່ໃນຖານຂໍ້ມູນ MySQL:CREATE INDEX index_id USINGB-TREE ON student(std_id);

1.4 ວິທີການສ້າງອິນເດັກ Bitmap

ຕົວຢ່າງ: ຄຳສັ່ງ SQL ແບບ Bitmap ທີ່ຢູ່ ໃນຖານຂໍ້ມູນ MySQL: CREATE FULLTEXT index_id ON student(std_id);

1.5 ການທົດລອງ

ໃນການທົດລອງ ເຮົາໄດ້ທົດລອງໃນຮູບແບບທີ່ບໍ່ມີການສ້າງອິນເດັກກ່ອນ. ຫຼັງຈາກນັ້ນເພີ່ມປະສິດທິພາບໃນການສ້າງອິນເດັກໃນຮູບແບບ Bitmap ແລະ B-Tree ເພື່ອໃຫ້ເຫັນເຖິງຄວາມແຕກຕ່າງລະຫວ່າງການຄົ້ນຫາຂໍ້ມູນທີ່ບໍ່ມີການສ້າງ ອິນເດັກອິນເດັກ ແລະ ພາຍຫຼັງມີການສ້າງ ອິນເດັກ.



ຮູບທີ 2. ຜົນຂອງການທົດລອງ

2. ຜົນໄດ້ຮັບ

ພວກເຮົານຳສະເໜີຜົນຂອງການທົດລອງໃນການເພີ່ມປະສິດທິພາບໃນການຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາ ທັງໝົດມີ 4 ກໍລະນີ ດັ່ງລຸ່ມນີ້:

Number of rows	Output Time (s)		
	Without indexing	Bitmap index	B-tree index
1000 rows	0.00697275	0.00622675	0.00693075
1500 rows	0.018467	0.008487	0.01005025
2000 rows	0.00946425	0.00731525	0.00829925
2500 rows	0.01226675	0.009699	0.0104335

ຮູບທີ 4. ເວລາຄົ້ນຫາຂໍ້ມູນນັກສຶກສາ

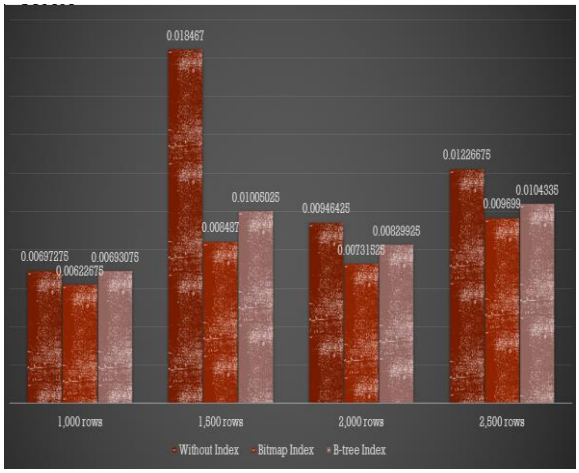
ກໍລະນີທີ 1 ຄື: ການປຽບທຽບປະສິດທິພາບການຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາໂດຍທີ່ບໍ່ມີການສ້າງອິນເດັກ ແລະ ມີການສ້າງອິນເດັກໃນຮູບແບບ Bitmap ແລະ B-Tree ສຳລັບຂໍ້ມູນທີ່ເອົາມາທົດລອງມີ 1000, 1500, 2000, ແລະ 2500 ແຖວ. ເຊິ່ງຜົນການທົດລອງເຫັນວ່າ: ການຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາແບບ Bitmap ໄວກ່ວາ ແບບ B-Tree 0.000704 ວິນາທີ ແລະ ໄວກວ່າແບບບໍ່ມີອິນເດັກ 0.000746 ວິນາທີ.

ກໍລະນີທີ 2 ຄື: ປຽບທຽບປະສິດທິພາບການຄົ້ນຫາຄະແນນຂອງນັກສຶກສາໂດຍທີ່ບໍ່ມີການສ້າງ. ຂໍ້ມູນທີ່ເອົາມາທົດລອງແມ່ນ 1000, 1500, 2000 ແລະ 2500 ແຖວເຫັນວ່າ: ການຄົ້ນຫາຄະແນນຂອງນັກສຶກສາແບບ Bitmap ໄວກ່ວາ B-Tree 0.000571 ວິນາທີ ແລະ ໄວກວ່າແບບບໍ່ມີ ອິນເດັກ 0.010179 ວິນາທີ.

ກໍລະນີທີ 3 ຄື: ການປຽບທຽບປະສິດທິພາບການຄົ້ນຫານັກສຶກສາຜູ້ທີ່ລົງທະບຽນເຊິ່ງຂໍ້ມູນທີ່ເອົາມາທົດລອງ 1000, 1500, 2000 ແລະ 2500 ເຮັກຄອດເຫັນວ່າ: ໃນຮູບແບບ Bitmap ໄວກ່ວາ B-Tree 0.000984 ວິນາທີ ແລະ ໄວກວ່າແບບບໍ່ມີ ອິນເດັກ 0.00249 ວິນາທີ.

ກໍລະນີທີ 4 ຄື: ປຽບທຽບປະສິດທິພາບການຄົ້ນຫາມລາຍວິຊາຂອງນັກສຶກສາຂໍ້ມູນທີ່ເອົາມາທົດລອງແມ່ນ 1000, 1500, 2000 ແລະ 2500 ແຖວ ເຫັນວ່າ: ການຄົ້ນຫາລາຍວິຊາຮຽນແບບ Bitmap

ໄວກ່ວາ ແບບ B-Tree 0.003040 ວິນາທີ ແລະ ໄວກ່ວາແບບບໍ່ມີ ອິນເດັກ 0.010484 ວິນາທີ.



ຮູບທີ 5. ເວລາການຄົ້ນຫາຂໍ້ມູນນັກສຶກສາ

3. ວິພາກຜົນ

ຜົນຂອງການທົດລອງໃນເຮົາສາມາດຕີລາຄາໄດ້ ດັ່ງນີ້:

ການຄົ້ນຫາຂໍ້ມູນຂອງນັກສຶກສາ,ຄະແນນຂອງນັກສຶກສາ,ລາຍງານການຈ່າຍເງິນຂອງນັກສຶກສາ ແລະ ນັກສຶກສາທີ່ລົງທະບຽນຮຽນລາຍວິຊາ ແມ່ນເໝາະສົມກັບແບບ ອິນເດັກ Bitmap.

ຈາກການທົດລອງເຫັນວ່າ ໃນກໍລະນີທີ່ບໍ່ໃຊ້ອິນເດັກການຄົ້ນຫາຂໍ້ມູນຈະບໍ່ມີປະສິດທິພາບເທົ່າກັນ ກັບໃນກໍລະນີທີ່ມີການໃຊ້ ອິນເດັກ ແລະ ໃນກໍລະນີທີ່ໃຊ້ ອິນເດັກໃນຮູບແບບ Bitmap ແລະ B-Tree ຈະເຫັນໄດ້ວ່າການຄົ້ນຫາຂໍ້ມູນ ຖ້າຫາກວ່າຂໍ້ມູນມີຈຳນວນ ໜ້ອຍແມ່ນການຄົ້ນຫາແບບ B-Tree ໄວກວ່າ ແບບ Bitmap ແລະ ຖ້າຂໍ້ມູນມີຈຳນວນຫຼາຍ ການຄົ້ນຫາຂໍ້ມູນໃນຮູບແບບ Bitmap ຈະໄວກວ່າ ແບບ B-Tree ແລະ ຄວາມແຕກຕ່າງໃນການຄົ້ຫາຂໍ້ມູນທັງສອງແບບນີ້ຈະມີຄວາມແຕກຕ່າງກັນເລັກນ້ອຍ.

4. ສະຫຼຸບຜົນ

ການເພີ່ມປະສິດທິພາບໃນການຄົ້ນຫາຂໍ້ມູນນັກສຶກສາຂອງມະຫາວິທະຍາໄລສຸພານຸວົງ ຈະເຫັນໄດ້ວ່າການສ້າງອິນເດັກສາມາດເພີ່ມປະສິດທິພາບໃນການຄົ້ນຫາຂໍ້ມູນ ແລະ ສາມາດຊ່ວຍຫຼຸດຜ່ອນເວລາໃນການຄົ້ນຫາຂໍ້ມູນໄດ້. ແຕ່ການທີ່ຈະໄດ້ປະສິດທິພາບນັ້ນຈຳເປັນຕ້ອງມີຄວາມຮູ້ຄວາມເຂົ້າໃຈຂອງການປະມວນຜົນການຄົ້ນຫາຂໍ້ມູນ ເພື່ອທີ່ຈະເລືອກໃຊ້ປະເພດຂອງ ອິນເດັກໃຫ້ຖືກຕ້ອງກັບກໍລະນີ ຫຼື ຮູບແບບການຄົ້ນຫາຂໍ້ມູນໂດຍຈະສິ່ງຜົນຕໍ່ປະສິດທິພາບ ແລະເວລາໃນການຄົ້ນຫາຂໍ້ມູນລວມເຖິງປະສິດທິພາບໂດຍລວມຂອງລະບົບຈັດການຖານຂໍ້ມູນຕື່ມ. ດັ່ງນັ້ນ ການສ້າງອິນເດັກນັ້ນ ນອກຈາກເປັນການເພີ່ມປະສິດ ທິພາບ ແລະ ຫຼຸດຜ່ອນເວລາການໃນການຄົ້ນຫາຂໍ້ມູນແລ້ວຍັງເປັນການຫຼຸດຜ່ອນການເຮັດວຽກຂອງລະບົບຈັດການຖານຂໍ້ມູນອີກດ້ວຍ.

ຈາກການປະຕິບັດການທົດລອງ ແລະ ວິໄຈໃນຄັ້ງນີ້ ສະແດງໃຫ້ເຫັນວ່າການນຳໃຊ້ອິນເດັກ Bitmap ແລະ ອິນເດັກ B-tree ແມ່ນໄດ້ເພີ່ມປະສິດທິພາບໃນການຄົ້ນຫາຂໍ້ມູນໄດ້ຢ່າງແທ້ຈິ່ງ,ໂດຍສະເພາະແມ່ນຄົ້ນຫາໃນຖານຂໍ້ມູນທີ່ມີຂໍ້ມູນຫຼາກຫຼາຍ.

ຈາກ ຜົນຂອງການວິໄຈຂ້າງເທິງນີ້ ພວກເຮົາຍັງສາມາດທົດສອບການເພີ່ມປະສິດທິພາບໃນການຄົ້ນຫາຂໍ້ມູນໂດຍການສ້າງອິນເດັກຄື:

- ການທົດສອບໃນລະບົບຈັດການກັບຖານຂໍ້ມູນທີ່ແຕກຕ່າງຈາກ MySQL ເພາະວ່າໃນແຕ່ລະລະບົບຈັດການຖານຂໍ້ມູນຈະມີການຈັດການກັບຂໍ້ມູນທີ່ແຕກຕ່າງກັນ.
- ທົດສອບກັບລະບົບປະຕິບັດການອື່ນ ເຊັ່ນວ່າ: Linux.

5. ຄວາມຮູ້ບຸນຄຸນ

ຂ້າພະເຈົ້າຫຂໍສະແດງຄວາມຮູ້ບຸນຄຸນມາຍັງຄະນະນຳຄະນະວິສະວະກຳສາດ,ຫ້ອງການຄົ້ນຄ້ວາ ແລະ ບໍລິຫານວິຊາການ ແລະ ມະຫາວິທະຍາໄລສຸພານຸວົງ, ຂໍຂອບໃຈມາຍັງ Prof. Guowu Yang ທີ່ໃຫ້ຄຳປຶກສາ ແລະ ຄວາມຄິດເຫັນຕ່າງໆທີ່ປະກອບຢູ່ໃນບົດວິໄຈນີ້ທີ່ໃຫ້ຄວາມສະດວກ ແລະ ໄດ້ທົດລອງໃນຄັ້ງນີ້ຈົນສຳເລັດດ້ວຍດີ

6. ເອກະສານອ້າງອີງ

[1] Tongkaw, S., & Tongkaw, A. (2016). A comparison of database performance of MariaDB and MySQL

with OLTP workload. In *2016 IEEE conference on open systems (ICOS)* (pp. 117-119).

[2] Rajurkar, J., & Khan, T. K. (2015). Efficient query processing and optimization in SQL using compressed bitmap indexing for set predicates. In *2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO)* (pp. 1-5).

[3] Rosnan, S., Abd Rahman, N., Hatim, S. M., & Ghul, Z. H. (2019). Performance evaluation of inverted files, B-Tree and B+ Tree indexing algorithm on Malay text. In *2019 4th International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE)* (pp. 1-6).

[4] Sainui, J., Vanichayobon, S., & Wattanakitrungroj, N. (2008, December). Optimizing encoded bitmap index using frequent itemsets mining. In *2008 International Conference on Computer and Electrical Engineering* (pp. 511-515).

[5] Lin, H. Y., & Chen, S. Y. (2009). Indexing and Querying in Multimedia Databases. In *2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing* (pp. 475-478).

[6] Frühwirt, P., Huber, M., Mulazzani, M., & Weippl, E. R. (2010). InnoDB database forensics. In *2010 24th IEEE International Conference on Advanced Information Networking and Applications* (pp. 1028-1036).

[7] Dong, X., & Li, X. (2015). A novel distributed database solution based on MySQL. In *2015 7th International Conference on Information Technology in Medicine and Education (ITME)* (pp. 329-333).

[8] Ni, J., Xie, S., Li, Z., & Jia, C. (2018). Optimization Design Method of Cache Extension in MySQL Database. In *2018 IEEE International Conference on Mechatronics and Automation (ICMA)* (pp. 2295-2299).

[9] Al-Kandari, A., & Alhouli, M. (2014). Fuzzy Object Relational Database Management System (FORDBMS) is appropriate approach for Real-Estate (GIS) business. In *2014 Fourth International Conference on Digital Information and Communication Technology and its Applications (DICTAP)* (pp. 114-117).

[10] Feng, D., Zhang, Z., Zhou, F., & Ji, J. (2008). Application study of data mining on customer relationship management in E-commerce. In *2008 9th International Conference on Computer-Aided Industrial Design and Conceptual Design* (pp. 706-710).

[11] Wang, K., Ooi, B. C., & Sung, S. Y. (1999). P-Tree: A B-Tree Index for Lists. In *Database Systems for Advanced Applications, International Conference on* (pp. 221-221).

[12] Naim, N. F., Yassin, A. I. M., Zamri, W. M. A. W., & Sarnin, S. S. (2011). Mysql Database for storage of fingerprint data. In *2011 UkSim 13th International Conference on Computer Modelling and Simulation* (pp. 293-298).

[13] Ahmad, A., Khan, N. U., & Abbas, A. W. (2013). PHP+ MySQL Based Online Examination System with Power Failure Handling and Dropbox Capability. In *2013 IEEE Seventh International Conference on Software Security and Reliability Companion* (pp. 21-25).

[14] Matthews, M., Cole, J., & Gradecki, J. D. (2003). *MySQL and Java developer's guide*. John Wiley & Sons.

[15] Ongo, G., & Kusuma, G. P. (2018). Hybrid database system of mysql and mongodb in web application development. In *2018 International Conference on Information Management and Technology (ICIMTech)* (pp. 256-260).

[16] Comer, D. (1979). Ubiquitous B-tree. *ACM Computing Surveys (CSUR)*, *11*(2), 121-137.

[17] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). Greedy algorithms. *Introduction to algorithms*, *1*, 329-355.

[18] Mohan, C., & Narang, I. (1992). Algorithms for creating indexes for very large tables without quiescing updates. In *Proceedings of the 1992 ACM SIGMOD international conference on Management of data* (pp. 361-370).

[19] Basra, M. K. A., Salek, M. S., Camilleri, L., Sturkey, R., & Finlay, A. Y. (2015). Determining the minimal clinically important difference and responsiveness of the Dermatology Life Quality Index (DLQI): further data. *Dermatology*, *230*(1), 27-33.

[20] Sharma, S., Sharma, M., Jain, R., & Khatri, S. K. (2017). Nlcs based string approximation for searching indexing keywords in b-tree. In *2017 2nd*