Sinh viên thực hiện:

19126053-Dương Quang Minh

19126044-Lê Trung Hiếu

19126054-Nguyễn Gia Minh

18126020- Trần Bảo Khánh

GV phụ trách: TS THÁI LÊ VINH

HỒ THỊ HOÀNG VY

Đồ án/bài tập môn học  - CƠ SỞ DỮ LIỆU NÂNG CAO

HỌC KỲ III – NĂM HỌC 2021-2022

**môn cơ sở dữ liỆu nâng cao**

**BỘ MÔN HỆ THỐNG THÔNG TIN – KHOA CÔNG NGHỆ THÔNG TIN**

**ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM, ĐẠI HỌC QUỐC GIA TP.HCM**

**BẢNG THÔNG TIN CHI TIẾT NHÓM**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Mã nhóm:** | **06** | | | |
| **Tên nhóm:** |  | | | |
| **Số lượng:** | **4** | | | |
| **MSSV** | **Họ tên** | **Email** | **Điện thoại** | **Hình ảnh** |
| 18126044 | Lê Trung Hiếu | 19126044@student.hcmus.edu.vn |  |  |
| 18126053 | Dương Quang Minh | 19126053@student.hcmus.edu.vn |  |  |
| 18126054 | Nguyễn Gia Minh | 19126054@student.hcmus.edu.vn |  |  |
| 18126020 | Trần Bảo Khánh | 18126006@student.hcmus.edu.vn |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Bảng phân công & đánh giá hoàn thành công việc** | | | |  |
| **Công việc thực hiện** | **Người thực hiện** | **Mức độ hoàn thành** | **Đánh giá của nhóm** |  |
| Đặc tả đồ án |  | 100% | 9/10 |  |
| Quy trình nghiệp vụ |  | 100% | 10/10 |  |
| Toàn vẹn dữ liệu |  | 100% | 10/10 |  |
| Yêu cầu chức năng |  | 100% | 10/10 |  |
| Xác định yêu cầu dữ liệu |  | 100% | 10/10 |  |
| Layout |  | 100% | 9/10 |  |
| Xác định chỉ mục |  | 100% | 10/10 |  |
| Cài đặt chỉ mục |  | 100% | 10/10 |  |
| Phân tích |  | 100% | 10/10 |  |
| Kiểm chứng hiệu quả của chỉ mục |  | 100% | 10/10 |  |
| Partition |  | 100% | 10/10 |  |
| Chức năng | | | |  |
| Trang admin   * Xem doanh thu * Xem danh sách user trong hệ thống, tìm kiếm theo tên * Xem danh sách, thông tin, tìm kiếm các nhân viên. |  | 100% | 10/10 |  |
| Chức năng của nhân viên:   * Xem thông tin của khách hàng và chỉnh sửa * Thêm thông tin, tạo tour mới vào hệ thống |  | 100% | 10/10 |  |
| Giỏ Hàng:   * Ước tính giá tour, thay đổi số lượng hoặc xóa * Chức năng thanh toán đơn hàng. |  | 100% | 10/10 |  |
| Trang chủ:   * Xem danh sách tour trên trang chủ * Hiển thị theo danh sách , tìm kiếm theo tên tour * Đăng nhấp , đăng ký |  | 100% | 10/10 |  |
| User:   * Xem thông tin cá nhân, thay đổi thông tin cá nhân * -Xem thông tin các tour, xem danh sách các tour đã đặt, lịch sử đặt tour |  | 100% | 10/10 |  |

**YÊU CẦU ĐỒ ÁN- BÀI TẬP**

|  |  |
| --- | --- |
| **Loại bài tập** | **🗹 Lý thuyết Thực hành Đồ án 🗹 Bài tập** |
| **Ngày bắt đầu** |  |
| **Ngày kết thúc** |  |

Contents

**[Yêu cầu của Đồ án/Bài tập](#_Toc80858561)** [4](#_Toc80858561)

**[1.](#_Toc80858562)****[Mô Tả:](#_Toc80858562)** [4](#_Toc80858562)

[a.Quy trình nghiệp vụ 4](#_Toc80858563)

*[1)Quy trình cho khách hàng:](#_Toc80858564)* [4](#_Toc80858564)

*[2)Quy trình cho nhân viên (Bao gồm cả admin lẫn nhân viên thông thường):](#_Toc80858565)* [4](#_Toc80858565)

*[3)Quy trình thanh toán:](#_Toc80858567)* [4](#_Toc80858567)

b.Danh sách chức năng

**[2.](#_Toc80858569)****[Thiết kế dữ liệu quan niệm](#_Toc80858569)** [4](#_Toc80858569)

[a. Đặc tả hệ thống 4](#_Toc80858570)

[b. Ràng buộc dữ liệu 6](#_Toc80858571)

**[3. Thiết kế dữ liệu logic](#_Toc80858572)** [11](#_Toc80858572)

[a. Mô hình ER 11](#_Toc80858573)

[b. Mô hình quan hệ 12](#_Toc80858574)

[c. Chuẩn hóa dữ liệu: 13](#_Toc80858575)

**[4. Thiết kế vật lý (thông tin tần suất, cài chỉ mục, phân tích, kiểm chứng hiệu quả chỉ mục…)](#_Toc80858576)** [16](#_Toc80858576)

[a. Thông tin tần suất 16](#_Toc80858577)

[b. Chỉ mục 16](#_Toc80858578)

[c. Phân tích 18](#_Toc80858579)

[d. Kiểm chứng hiệu quả chỉ mục 19](#_Toc80858580)

[e. Trigger và Procedure 36](#_Toc80858581)

**[5. Xác định yêu cầu chức năng](#_Toc80858582)**[: 42](#_Toc80858582)

**[6.](#_Toc80858583)****[Layout của giao diện:](#_Toc80858583)** [45](#_Toc80858583)

**[7.](#_Toc80858584)****[Partiton](#_Toc80858584)** [56](#_Toc80858584)

**[I.](#_Toc80858585)****[Partition:](#_Toc80858585)** [56](#_Toc80858585)

**[II.](#_Toc80858586)****[Lợi ích của việc sử dụng Partition:](#_Toc80858586)** [57](#_Toc80858586)

**[III.](#_Toc80858587)****[Các kiểu phân chia dữ liệu:](#_Toc80858587)** [57](#_Toc80858587)

**[IV.](#_Toc80858588)****[Phân chia dữ liệu bảng Order (Hoá đơn) theo chiều ngang](#_Toc80858588)** [57](#_Toc80858588)

**Mô tả**

**A/Quy trình nghiệp vụ**

1. *Quy trình cho người mua:*

* Đăng nhập tài khoản.
* Tìm kiếm, tham khảo thông tin tour, dịch vụ, khuyến mại.
* Người Mua đưa ra quyết định đặt tour bằng cách click vào “Đặt hàng”.
* Người Mua lựa chọn cách thức giao dịch tuy theo chính sách bên bán.
* Đơn hàng của Người Mua sẽ được chuyển thông tin đến bên bán.
* Người Mua nhận được thông tin xác nhận về đơn hàng
* Người Mua chỉnh sửa đơn hàng.

1. *Quy trình cho bên cung cấp tour:*

* Đăng nhập tài khoản
* Tạo tài khoản (chỉ dành cho nhân viên có quyền admin)
* Tìm kiếm, xem thông tin, cập nhật, tạo các tour
* Quản lý đơn hàng
* Sửa bảng trạng thái của tour
* Thống kê thông tin
* Gửi thông báo đến người mua
  1. *Quy trình thanh toán:*

**B/Danh sách yêu cầu chức năng**

* Đăng nhập/Đăng ký user
* Tìm lại password
* Hiển thị sản phẩm
* Thêm sản phẩm vào giỏ hàng
* Thêm/xóa/ cập nhật thông tin sản phẩm
* Order sản phẩm
* Chỉnh sửa thông tin user
* Sắp xếp hiển thị sản phẩm theo loại sản phẩm

1. Yêu cầu phi chức năng

* Toàn vẹn dữ liệu
* Khả năng tương tác
* Khả năng sử dụng
* Khả năng quản lí
* Khả năng bảo trì

**Thiết kế dữ liệu quan niệm**

**A/.Đặc tả hệ thống**

Công ty MHMK mở một trang buôn bán, quản lý các tour du lịch. Công ty cần xây dựng 1 trang web cho phép công ty và các cá nhân thực hiện việc mua bán các tour.

Sau đây là kết quả của việc phân tích yêu cầu ứng dụng:

* Trang MHMKTour bán đủ loại tour với các tour do công ty tự bán và quản lý.
* Một **nhân viên**  có: *mã nhân viên, tên tài khoản, tên nhân viên*, với nhân viên không phải là admin, thì có thêm thông tin về Admin quản lý nhân viên đó. Với nhân viên có cấp bậc là admin có quyền tạo các tài khoản nhân viên và quy định cấp bậc cho họ. Với các nhân viên còn lại có quyền thay đổi, cập nhật, tạo các tour.
* Cần lưu trữ **thông tin khách hàng**, gồm : *mã khách hàng, tên khách hàng, số điện thoại,địa chỉ, email .*Mỗi khách hàng chỉ được sử dụng 1 số điện thoại duy nhất để đăng ký.
* Cần lưu lại thông tin về tất cả các tour mà công ty MHMK cung cấp gồm: *Mã tour, ngày bắt đầu, ngày kết thúc, mô tả, nơi bắt đầu, nơi tham quan, phương tiện di chuyển, giá thành*. Mỗi tour có một nơi bắt đầu tour gồm : *mã địa điểm, tên nơi bắt đầu*. Ngoài ra, các tour còn có nơi tham quan gồm: *mã địa điểm, tên nơi tham quan*. .
* Khi khách hàng chọn tour, thì sản phẩm sẽ được đưa vào **giỏ hàng** của khách hàng đó, có: Mã đơn, Mã khách hàng và Tổng tiền, bên cạnh đó, các các thông tin chi tiết của đơn hàng sẽ được lưu riêng biệt bao gồm: Mã tour, số lượng người lớn, số lượng trẻ em. Ngay sau khi khách hàng đặt mua thì trạng thái của đơn hàng sẽ được chuyển thành *chưa thanh toán.* Sau khi nhận được thông tin đơn hàng, nhân viên sẽ căn cứ vào số điện thoại của khách hàng được lưu trên thông tin thanh toán sẽ gọi để trao đổi thêm thông tin với khách hàng và thông tin tới khách hàng về việc thanh toán. Khi xác nhận thông tin thanh toán của khách hàng, nhân viên sẽ chuyển trạng thái của đơn hàng thành *đã thanh toán,* Tuy nhiên, nếu như khách hàng hủy hoặc quá thời hạn giữ đơn hàng mà khách hàng chưa thanh toán, đơn sẽ được hủy, nhưng để đảm bảo việc phục hồi đơn khi cần thiết, đơn hàng sẽ không được xóa hoàn toàn mà sẽ được đưa ra khỏi các tiến trình xử lý.
* MHMKTour cần lưu lại các thông tin giao dịch vào **nhật kí giao dịch** và lưu trữ theo: *mã giao dịch, mã đơn hàng, ngày lập giao dịch.* Tuy nhiên, chỉ các đơn hàng có trạng thái là *đã thanh toán* thì mới được đưa vào nhật ký giao dịch.

**B/.Ràng buộc dữ liệu**

* **Admin:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdAdmin | int | PRIMARY Key |
| Username | nvarchar(50) | NOT NULL |
| Password | nvarchar(50) | NOT NULL |

* **Staff:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdStaff | int | PRIMARY Key |
| IdAdmin | int | FOREIGN KEY |
| Fullname | nvarchar(50) | NOT NULL |
| Username | nvarchar(50) | NOT NULL |
| Password | nvarchar(50) | NOT NULL |

* **Client:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdClient | int | PRIMARY Key |
| Username | nvarchar(32) | NOT NULL |
| Password | nvarchar(32) | NOT NULL |
| Fullname | nvarchar(10) | NOT NULL |
| Phone | nvarchar(10) | UNIQUE |
| Address | nvarchar(50) |  |
| Email | nvarchar(50) |  |

* **Tour:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdTour | int | PRIMARY Key |
| TourName | nvarchar(50) | UNIQUE |
| DayStart | Date | NOT NULL |
| DayEnd | Date | NOT NULL |
| Description | nvarchar(10) | NOT NULL |
| LocationStart | nvarchar(10) | FOREIGN KEY |
| Vehicle | nvarchar(50) | NOT NULL |
| Cost | float | NOT NULL |

* **Description:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdDescription | int | PRIMARY Key |
| Tour | int | FOREIGN Key |
| Total | float | NOT NULL |

* **DescriptionDetail:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdDescriptionDetail | int | PRIMARY Key |
| IdDescription | int | FOREIGN Key |
| nameDescription | nvarchar(50) | NOT NULL |
| Adults | int | NOT NULL |
| Childs | int | NOT NULL |
| Incurred | float |  |
| Total | float | NOT NULL |

* **OrderTour:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdOrder | int | PRIMARY Key |
| IdClient | int | FOREIGN Key |
| Total | float | NOT NULL |

* **OrderDetail:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | **int** | PRIMARY KEY |
| IdOrder | int | FOREIGN KEY |
| IdTour |  | FOREIGN KEY |
| Adults | int | NOT NULL |
| Childrens | int | NOT NULL |

* **LocationVisit:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | int | PRIMARY Key |
| VisitPlace | nvarchar | NOT NULL |
| idTour | char | FOREIGNKEY |

* **OrderStatus:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Id | int | PRIMARY KEY |
| IdOrder | int | FOREIGN KEY |
| Status | char | NOT NULL |

* **Trash:**

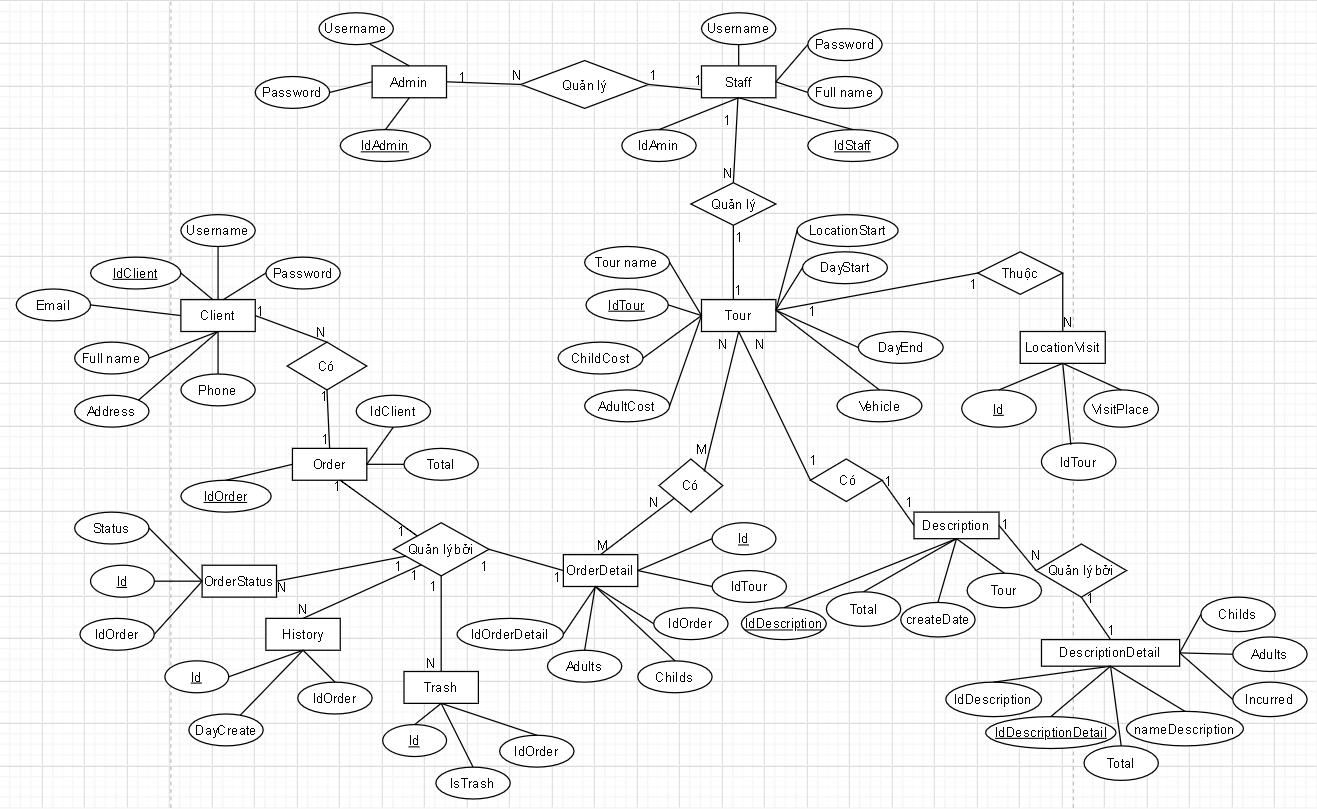
|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Id | int | PRIMARY KEY |
| IdOrder | int | FOREIGN KEY |
| IsTrash | boolean | NOT NULL |

* **History:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  |  |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Id | int | PRIMARY KEY |
| IdOrder | int | FOREIGN KEY |
| CreateDate | DateTime | NOT NULL |

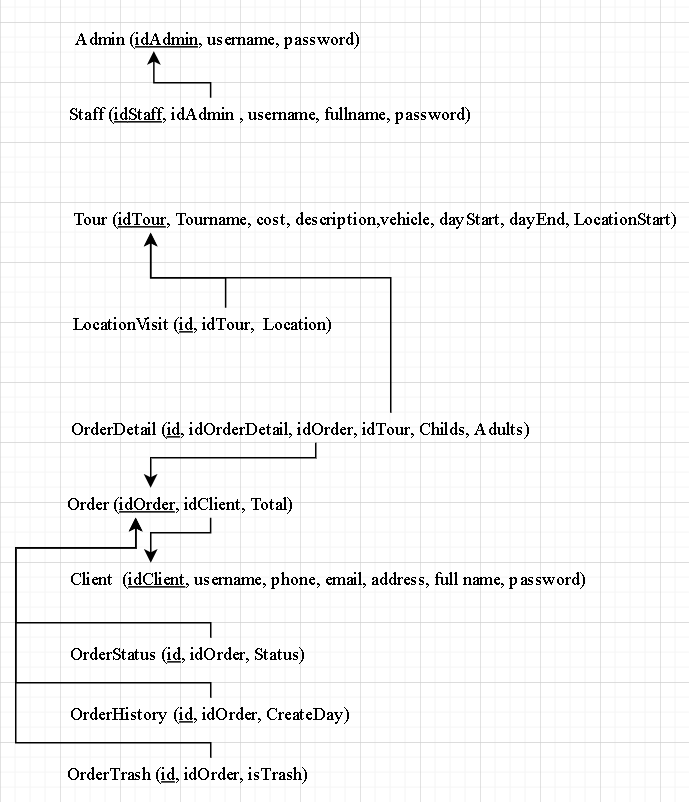
**Thiết kế dữ liệu logic**

1.Mô hình ER



Mô hình ER được xây dựng dựa trên cách thức vết dầu loang, với trung tâm là Tour.

1. Mô hình quan hệ



3.Chuẩn hóa dữ liệu:

* Client

Client  (idClient, username, phone, email, address, full name, password) không thuộc dạng chuẩn.

Trong bảng này có một trường là thuộc tính đa trị : address

* + Bảng mới sau khi nâng chuẩn sẽ là

Client (idClient, username, phone, email, full name, password)

Address (id, idClient, number, street, wards, district, city )

* + Bảng Customer sẽ ở dạng chuẩn 3NF :
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Tour

Tour (idTour, Tourname, cost, description,vehicle, dayStart, dayEnd, LocationStart)

Trong bảng này có hai trường là thuộc tính đa trị : Vehicle, LocationStart

* + Bảng mới sau khi nâng chuẩn sẽ là :

Tour (idTour, Tourname, cost, description,IdVehicle, dayStart, dayEnd)

LocationStart(id, idTour, Location)

Vehicle(id, idTour, Vehicle, Capacity)

* + Bảng Customer sẽ ở dạng chuẩn 3NF :
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Admin

Admin (idAdmin, username, password)

* + Bảng Admin sẽ ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Staff

Staff (idStaff, username, fullname, password)

* + Bảng Staff sẽ ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* LocationVisit

LocationVisit (id, idTour, Location)

* + Bảng Location sẽ ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* Order

Order (idOrder, idClient, Total)

* + Bảng Order sẽ ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* OrderDetail

OrderDetail (id, idOrderDetail, idOrder, idTour, Childs, Adults)

* + Bảng OrderDetail sẽ ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* OrderStatus

OrderStatus (id, idOrder, Status)

* + Bảng OrderStatus sẽ ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* OrderHistory

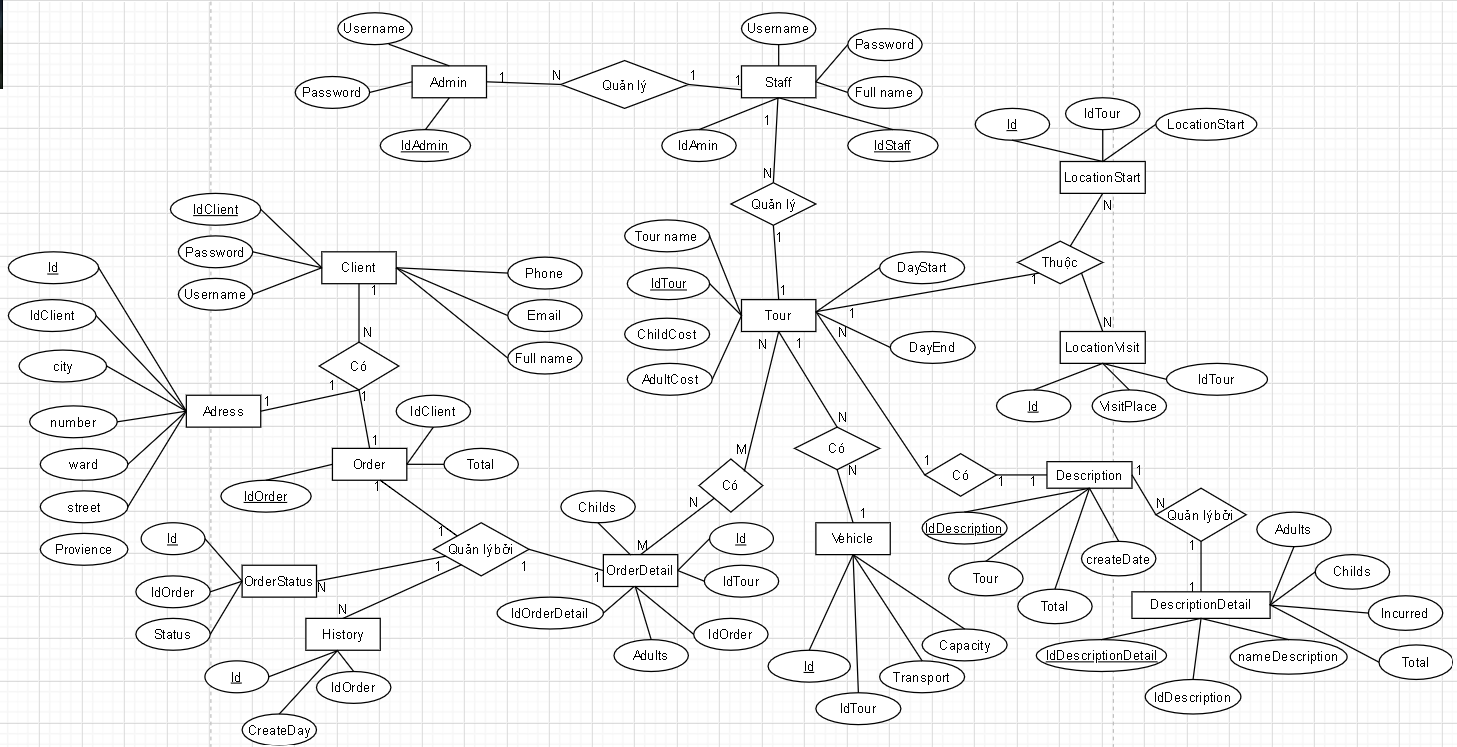
OrderHistory (id, idOrder, CreateDay)

* + Bảng Admin sẽ ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu
* OrderTrash

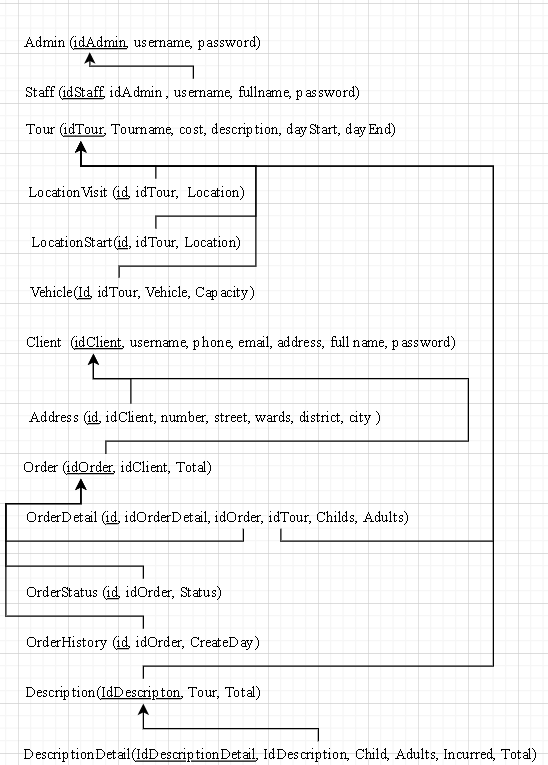
OrderTrash (id, idOrder, isTrash)

* + Bảng Admin sẽ ở dạng chuẩn 3NF:
    - Là 2NF
    - Không có phụ thuộc hàm bắc cầu

1. ERD, lược đồ quan hệ và miêu tả các trường dữ liệu sau khi được chuẩn hóa.
2. Mô hình ERD



1. Lược đồ quan hệ



1. Mô tả các trường dữ liệu bổ sung

* **Tour:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdTour | int | PRIMARY Key |
| TourName | nvarchar(50) | UNIQUE |
| DayStart | Date | NOT NULL |
| DayEnd | Date | NOT NULL |
| Description | nvarchar(10) | NOT NULL |
| Cost | float | NOT NULL |

* **LocationStart:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdLocation | int | PRIMARY Key |
| Provience | nvarchar | NOT NULL |
| Tour | char | FOREIGNKEY |

* **Vehicle:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| Id | char | PRIMARY Key |
| Vehicle | char | NOT NULL |
| Capacity | int | NOT NULL |
| Tour | char | FOREIGNKEY |

* **Client:**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| IdClient | int | PRIMARY Key |
| Username | nvarchar(32) | UNIQUE |
| Password | nvarchar(32) | NOT NULL |
| Fullname | nvarchar(10) | NOT NULL |
| Phone | nvarchar(10) | UNIQUE |
| Email | nvarchar(50) |  |

* **Address**

|  |  |  |
| --- | --- | --- |
| **Mô tả** |  | |
| **Dữ liệu** | **Kiểu dữ liệu** | **Ràng buộc** |
| id | int | PRIMARY Key |
| username | int | FOREIGNKEY |
| number | int | NOT NULL |
| street | Nvarchar(32) | NOTNULL |
| ward | Nvarchar(32) | NOTNULL |
| city | Nvarchar(32) | NOTNULL |
| provience | Nvarchar(32) | NOTNULL |

**Thiết kế vật lý**

**(thông tin tần suất, cài chỉ mục, phân tích, kiểm chứng hiệu quả chỉ mục…)**

1. Thông tin tần suất

|  |  |  |
| --- | --- | --- |
| **Chức năng** | **Mục tiêu** | **Tuần suất (lượt truy vấn)** |
| Tìm kiếm thông tin khách hàng | Tìm theo tên hoặc số điện thoại của khách hàng | 1M+/ ngày |
| Tìm kiếm thông tin nhân viên (Staff) | Tìm theo tên hoặc tên tài khoản | 10K/ngày |
| Tìm kiếm thông tin nhân viên (Admin) | Tìm theo tên tài khoản | 10K/ngày |
| Tìm kiếm thông tin tour | Tìm theo tên tour hoặc tìm theo những nơi tham quan | 10M+/ ngày |
| Hiển thị thông tin giỏ hàng | Hiển thị giỏ hàng theo mã khách hàng | 500k/ ngày |
| Hiển thị đơn hàng | Hiển thị theo mã khách hàng hoặc ngày tạo order | 500k/ngày |
| Hiển thị lịch sử giao dịch của KH | Tìm theo mã khách hàng và ngày tạo log | 500K/ ngày |
| Hiển thị trạng thái của các đơn hàng | Tìm theo tình trạng đơn hàng | 500K/ ngày |
| Hiển thị doanh thu mà ncc thu được | Tìm kiếm theo ngày tạo và mã Order | 500K/ ngày |
| Hiển thị mức chi mà ncc phải chi | Tìm kiếm theo mã Order, ngày tạo, thông tin, trạng thái tour | 500K/ ngày |

1. **Chỉ mục**

-- Index cho Admin

CREATE INDEX TimKiemAdmin ON Admin(Username ASC)

-- Index cho nhân viên

CREATE NONCLUSTERED INDEX TimKiemNhanVien ON Staff(

Fullname ASC,

Username ASC)

--Index cho khách hàng

CREATE NONCLUSTERED INDEX TimKiemKhachHang ON Client

(Fullname ASC,

Username ASC,

Phone ASC,

Email ASC)

-- Index cho danh sách Tour

CREATE NONCLUSTERED INDEX TimKiemTour ON Tour (TourName ASC, DayStart ASC, DayEnd ASC)

-- Index cho nơi bắt đầu

CREATE INDEX TimKiemLocationStart\_byLocation ON LocationStart (Province ASC)

CREATE INDEX TimKiemLocationStart\_byTour ON LocationStart (Tour ASC)

-- Index cho nơi tham quan

CREATE INDEX TimKiemLocationVisit\_byLocation ON LocationVisit (VisitPlace ASC)

CREATE INDEX TimKiemLocationVisit\_byTour ON LocationVisit (Tour ASC)

-- Index cho phương tiện

CREATE INDEX TimKiemPhuongTien ON Vehicle (Transport)

-- Index cho đơn hàng

CREATE NONCLUSTERED INDEX TimKiemOrder ON OrderTour(IdClient ASC, createDate ASC)

-- Index cho trang thái đơn hàng

CREATE NONCLUSTERED INDEX TimKiemOrderStatus ON OrderStatus (IdOrder ASC, OrderStatus ASC)

1. **Phân tích**
   * + Tìm kiếm thông tin nhân viên theo tên hoặc username: do tần suất tìm kiếm nhân viên để thực hiện công việc quản lý, nên ta cài index để cải thiện performance cho hệ thống
     + Tìm kiếm thông tin khách hàng theo tên, username hoặc số điện thoại: do tần suất tìm kiếm khách hàng để tạo order hoặc xem thông tin là cao, nên ta cài index để cải thiện performance cho hệ thống.
     + Tìm kiếm tour là chức năng có lượt truy vấn nhiều nhất trên hệ thống, vì vậy phải cải thiện performance cho việc truy vấn liên quan đến sản phẩm
     + Bên cạnh việc tìm kiếm tour dựa trên TourName, người dùng còn có thể tìm kiếm dựa trên phương tiện, nơi bắt đầu, nơi tham quan vì thế ta cài index ở những bảng Vehicle, LocationStart, LocationVisit để cải thiện performance cho việc truy vấn liên quan đến thông tin tour
     + Hiển thị đơn hàng: hiển thị theo id khách hàng hoặc ngày tạo order dùng để kiểm soát thông tin order của khách trong việc thanh toán hoặc kiểm tra doanh thu, lịch sử giao dịch của khách hàng.
     + Các bảng kiểm tra lịch sử giao dịch được admin và nhà cung cấp xem lại hàng ngày/tháng/năm để kiểm soát doanh thu và cải thiện việc bán hàng. Vì vậy cần cài đặt index ở các bảng này để việc truy suất dữ liệu được nhanh hơn (do dữ liệu lưu trữ rất lớn).
2. **Kiểm chứng hiệu quả chỉ mục**

Kích thước dữ liệu dùng kể kiểm chứng hiệu quả của chỉ mục

1. Admin : 1.000 dòng dữ liệu
2. Nhân viên (Staff): 150.000 nhân viên

3. Khách hàng : 500.000 khách hàng

4. Tour : 1.000.000 sản phẩm

5. Vehicle: 1.500.000 phương tiện

6. LocationStart: 1.000.000 vị trí

7. LocationVisit: 4.000.000 vị trí

8. Order: 800.000 đơn hàng

9. OrderStatus: 800.000 đơn hàng

Kết quả đánh giá một số Index (Index cho bảng Staff và bảng Client)

* Staff

Kiểm chứng hiệu quả việc sử dụng index dựa trên kết quả tìm kiếm nhân viên theo quản trị viên quản lý:

Query: Select \* from Staff where Admin = 'Admin667'

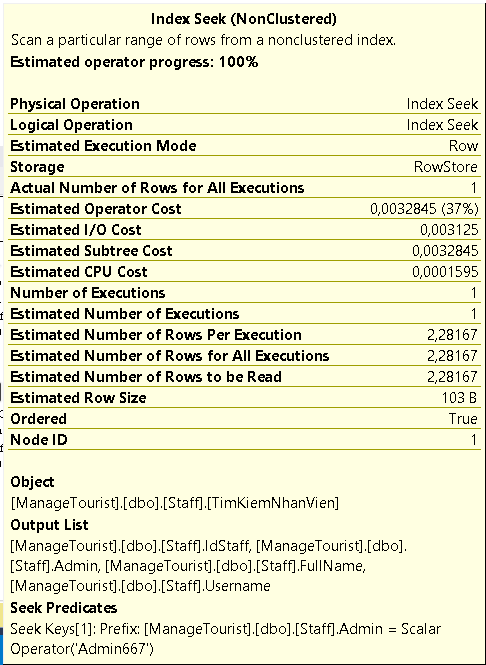
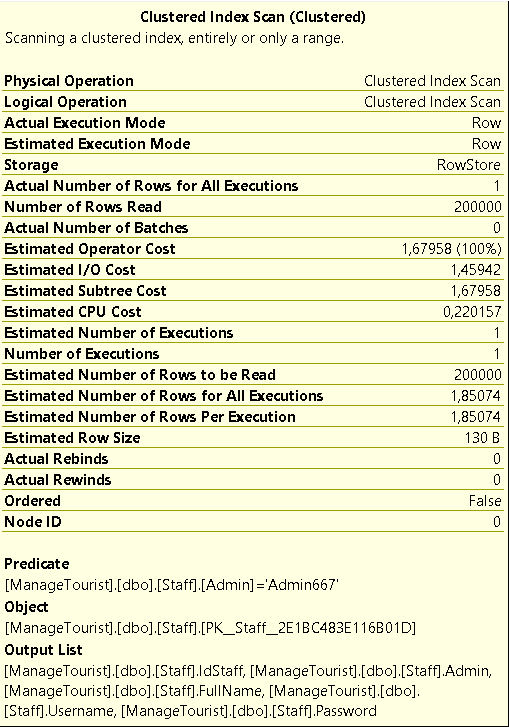
Kết quả:



Excution plan:

|  |  |
| --- | --- |
| Excution plan không dùng index | Excution plan dùng index |
|  |  |

Hiệu suất:

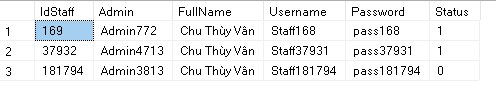


Sử dụng câu lệnh select cho bảng Staff ta thấy trước và sau khi sử dụng Index có sự cải thiện rõ rệt về hiệu quả của câu lệnh

Kiểm chứng hiệu quả việc sử dụng index dựa trên kết quả tìm kiếm nhân viên theo tên:

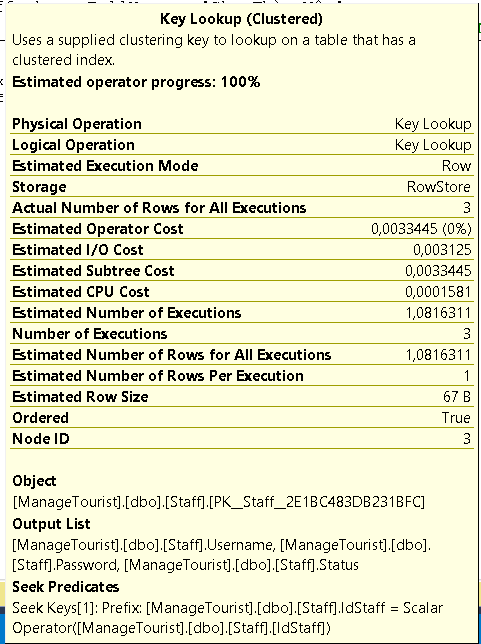
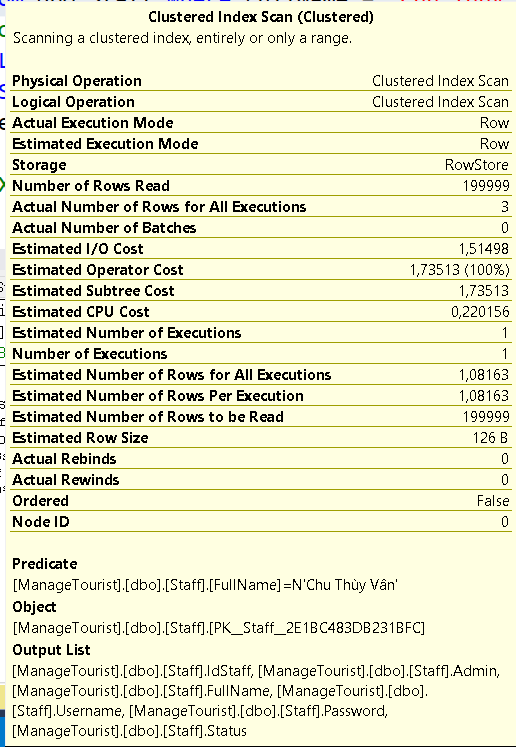
Query: Select \* from dbo.Staff where FullName = 'Chu Thùy Vân'

Kết quả



Excution plan:

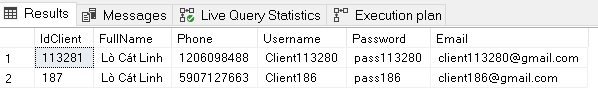
|  |  |
| --- | --- |
| Excution plan không dùng index | Excution plan dùng index |
|  |  |



* Client

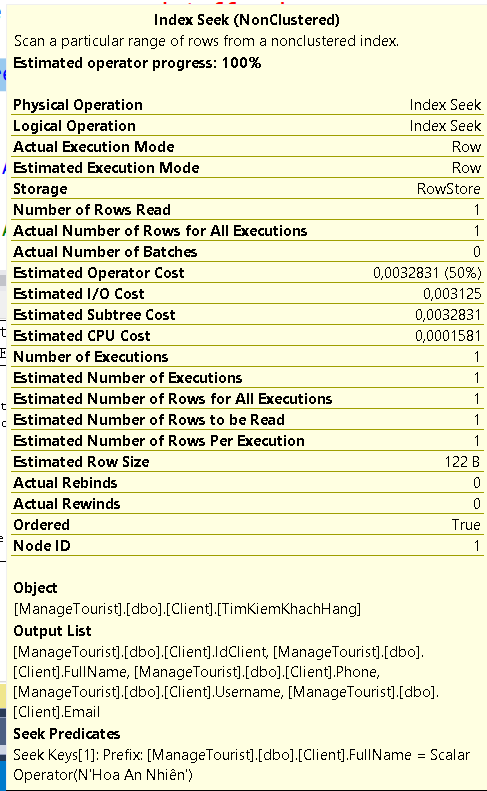
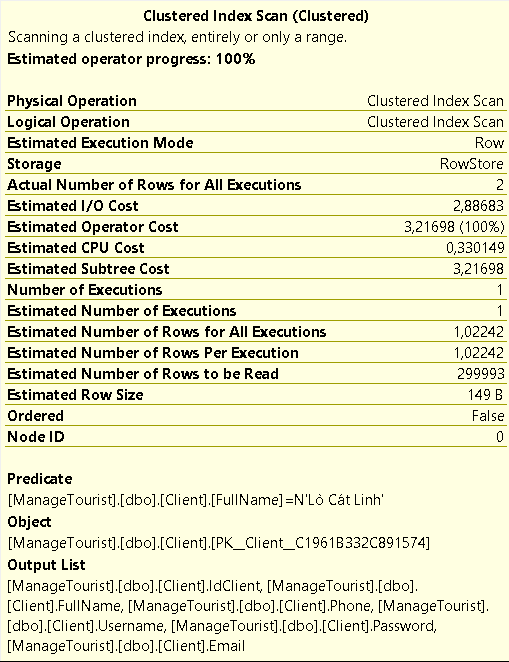
Query: Select \* from dbo.Client where FullName = 'Lò Cát Linh';

Kết quả:



Excution plan

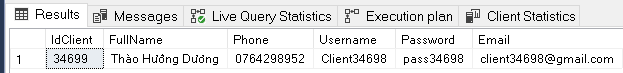
|  |  |
| --- | --- |
| Excution plan không dùng index | Excution plan với index |
|  |  |



Sử dụng câu lệnh select cho bảng Client ta thấy trước và sau khi sử dụng Index có sự cải thiện rõ rệt về hiệu quả của câu lệnh

Query: Select \* from dbo.Client where Email = ['client34698@gmail.com';](mailto:'client34698@gmail.com';)

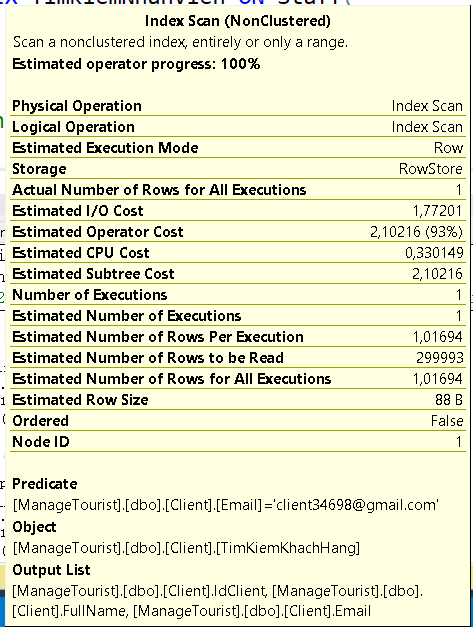
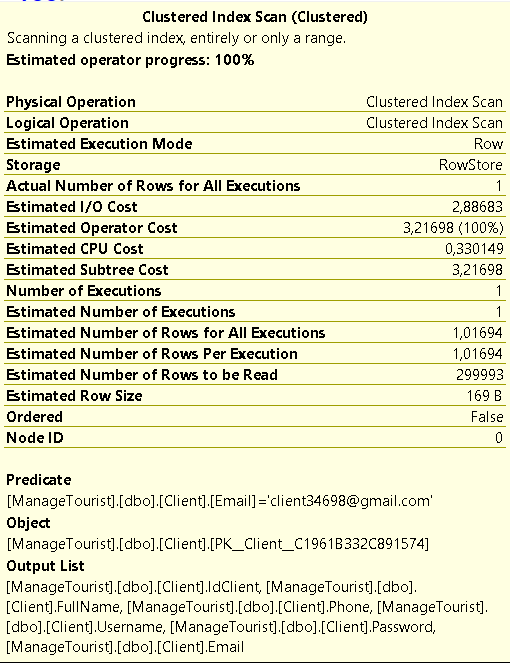
Kết quả:



Excution plan

|  |  |
| --- | --- |
| Excution plan không dùng index | Excution plan với index |
|  |  |

Hiệu suất:



***Đánh giá tổng:***

* Sử dụng index cho kết quả truy vấn tốt hơn, nhanh hơn so với khi không dùng.
* Cải thiện tốc độ khi phải kết nhiều bảng với nhau
* Khi sử dụng index thì các chi phí bỏ ra để truy vấn kết quả đã được giảm đi 1 nửa (sub tree, )
* Những chỉ số khác như I/O, Operator, CPU Cost được giảm đi rõ ràng và có thể thấy được.
* Vậy việc sử dụng index là cần thiết trong việc vận hành hiệu quả ứng dụng, hệ thống, cải thiện tốc độ cũng như hiệu suất truy xuất dữ liệu

1. **Trigger và Procedure**

* Trigger

CREATE TRIGGER trg\_OrderTour ON OrderDetail AFTER INSERT AS

BEGIN

UPDATE OrderTour

SET Total = (SELECT Adluts from OrderDetail where IdOrder = OrderDetail.IdOrder) \* (SELECT CostAdo from Tour where Tour = Tour.TourName) +

(Select Childs from OrderDetail where IdOrder = OrderDetail.IdOrder) \* (SELECT CostChild from Tour where Tour = Tour.TourName)

FROM OrderTour

JOIN inserted ON OrderTour.IdOrder = inserted.IdOrder

END

GO

CREATE TRiGGER trg\_CapNhatOrderTour ON OrderDetail After Update as

begin

UPDATE OrderTour

SET Total = (SELECT Adluts from OrderDetail where IdOrder = OrderDetail.IdOrder) \* (SELECT CostAdo from Tour where Tour = Tour.TourName) +

(Select Childs from OrderDetail where IdOrder = OrderDetail.IdOrder) \* (SELECT CostChild from Tour where Tour = Tour.TourName)

FROM OrderTour

JOIN deleted ON OrderTour.IdOrder = deleted.IdOrder

end

go

CREATE TRiGGER trg\_HuyOrderTour ON OrderDetail After delete as

begin

UPDATE OrderTour Set Total = 0

from OrderTour

JOIN deleted ON OrderTour.IdOrder = deleted.IdOrder

end

* Procedure

-- proc add Admin

create proc sp\_insert\_Admin

@username nvarchar(50),

@password nvarchar(50)

as

begin tran

begin try

insert into dbo.Admin values (@username, @password)

end try

begin catch

ROLLBACK TRANSACTION;

RETURN;

end catch

COMMIT TRANSACTION

GO

-- drop proc sp\_insert\_Admin

-- proc update Admin

create proc sp\_update\_Admin

@newusername nvarchar(50),

@newpassword nvarchar(50),

@oldusername nvarchar(50),

@oldpassword nvarchar(50),

@idAdmin int

as

begin tran

begin try

if exists (select \* from dbo.Admin where IdAdmin = @idAdmin and Username = @oldusername and Password = @oldpassword )

if (@newpassword is null)

update dbo.Admin set Username = @newusername where IdAdmin = @idAdmin

else if (@newusername is null)

update dbo.Admin set Password = @newpassword where IdAdmin = @idAdmin

else if (@newpassword is not null and @newusername is not null)

update dbo.Admin set Username = @newusername ,Password = @newpassword where IdAdmin = @idAdmin

else

raiserror ('Update information is null', 16, 1)

end try

begin catch

ROLLBACK TRANSACTION;

RETURN;

end catch

COMMIT TRANSACTION

GO

-- proc update Staff

create proc sp\_update\_Staff

@newusername nvarchar(50),

@newpassword varchar(50),

@oldusername nvarchar(50),

@oldpassword varchar(50),

@idStaff int

as

begin tran

begin try

if exists (select \* from dbo.Staff where IdStaff = @idStaff and Username = @oldusername and Password = @oldpassword )

if (@newpassword is null)

update dbo.Staff set Username = @newusername where IdStaff = @idStaff

else if (@newusername is null)

update dbo.Staff set Password = @newpassword where IdStaff = @idStaff

else if (@newpassword is not null and @newusername is not null)

update dbo.Staff set Username = @newusername ,Password = @newpassword where IdStaff = @idStaff

else

raiserror ('Update information is null', 16, 1)

end try

begin catch

ROLLBACK TRANSACTION;

RETURN;

end catch

COMMIT TRANSACTION

GO

-- proc add Staff

create proc sp\_insert\_Staff

@username nvarchar(50),

@password nvarchar(50),

@fullname nvarchar(50),

@admin nvarchar(50)

as

begin tran

begin try

if exists (select \* from dbo.Admin where dbo.Admin.Username = @admin)

insert into dbo.Staff values (@admin, @username, @password)

else

raiserror ('Failed Auth', 16, 1)

end try

begin catch

ROLLBACK TRANSACTION;

RETURN;

end catch

COMMIT TRANSACTION

GO

-- proc add Client

create proc sp\_insert\_Client

@FullName nvarchar(50),

@Phone varchar(50),

@Username varchar(50),

@hashPassword varchar(50),

@Email varchar(50),

@number int,

@street nvarchar(50),

@ward nvarchar(50),

@city nvarchar(50),

@provience nvarchar(50)

as

begin tran

begin try

insert into dbo.Client values (@FullName, @Phone, @Username, @hashPassword, @Email)

insert into dbo.UserAddress values (@Username, @number, @street, @ward, @city, @provience)

end try

begin catch

ROLLBACK TRANSACTION;

RETURN;

end catch

COMMIT TRANSACTION

GO

-- proc update cost

create proc dbo.Tour @newCostAdults float, @newCostChilds float, @idTour int

AS

begin tran

BEGIN Try

IF not exists (SELECT \* FROM dbo.Tour WHERE IdTour = @idTour)

raiserror ('Không tồn tại tour cần cập nhật', 16, 1)

else

begin

update dbo.Tour

set CostAdo = @newCostAdults, CostChild = @newCostChilds

where IdTour = @idTour

end

END Try

begin catch

ROLLBACK TRANSACTION;

RETURN;

end catch

COMMIT TRANSACTION

GO

**Xác định yêu cầu chức năng**

**I/ Chức năng:(dự kiến)**

1. Trang chủ:

* Xem danh sách các tour trên trang chủ.
* Hiển thị tour theo danh sách, tìm kiếm tour theo tên.
* Đăng nhập + đăng ký.
* Trang thông tin cá nhân:
* Xem thông tin cá nhân + địa chỉ (dùng để đón khi đã mua tour) + đơn hàng vừa đặt, thay đổi thông tin cá nhân.
* Thay đổi địa chỉ đón .
* Kiểm tra order theo mã order và email
* Xem danh sách tất cả order đã và đang đặt.

1. Admin:

* Xem các khoản thu - chi theo tháng, năm.
* Xem danh sách nhân viên có trong hệ thống (có thể thay đổi chức năng của người dùng ), tìm kiếm người dùng theo tên
* Xem danh sách khách hàng có trong hệ thống (có thể thay đổi chức năng của người dùng ), tìm kiếm người dùng theo tên

1. Nhân viên:

* Xem danh sách khách hàng có trong hệ thống (có thể thay đổi chức năng của người dùng ), tìm kiếm người dùng theo tên
* Thêm tour vào hệ thống
* Cập nhật thông tin tour.
* Xem danh sách đơn hàng và tùy chỉnh trạng thái đơn.
* Xem thu-chi theo ngày/ tháng/ năm, xem doanh thu từ ngày này sang ngày khác.
* Cập nhật các khoản chi của mỗi tour

1. Giỏ hàng

* Ước tính giá , thay đổi số lượng (số lượng tour, số lượng người đi) hoặc xóa cả giỏ hàng hoặc xóa 1 tour.
  + Chức năng thanh toán đơn.
* Chọn hình thức thanh toán, xong tiến hành thanh toán đơn hàng.

**II/ Usecase**

|  |  |  |
| --- | --- | --- |
|  | Đối tượng sử dụng - Trang có chức năng | Mô tả |
| Chức năng tạo đơn hàng mới, . Chỉnh sửa đơn hàng nếu chưa thực hiện thanh toán. | Khách hàng  ( Trang chủ - Giỏ hàng) | Khách hàng chọn tour cần đặt, tạo mới đơn, nhập số lượng người đi (bao gồm người lớn và trẻ em), chọn hình thức thanh toán. Nếu như khách hàng chưa đồng ý thanh toán thì có thể thay đổi tour, thay đổi số lượng người đi. |
| Chức năng đặt hàng chung | Khách hàng (Giỏ hàng) | Khi tiến hành đặt hàng, hệ thóng sẽ đưa ra phương thức thanh toán dự theo hình thức mà khách hàng đã chọn, sau đó nhân viên hệ thống sẽ xác nhận lại việc thanh toán và cập nhật trạng thái đơn của khách (đã thanh toán, chưa thanh toán, đang chờ thanh toán) |
| Chức năng cập nhật, thêm mới các tour | Nhân viên (Trang quản lý - phân hệ nhân viên) | Nhân viên có thể thêm các thông tin tour, cập nhật các tour |
| Chức năng theo dõi thu - chi, lịch sử các đơn hàng, tình trạng đơn hàng | Admin - Nhân viên (Trang quản lý - phân hệ Admin - Nhân viên) | Phía quản lý có thể xem thống kê doanh thu của tháng này/ năm/ tổng doanh thu., các khoản chi ở từng địa điểm của tour, tổng các khoản chi của một tour, hoặc của nhiều tour theo những mốc thời gian. Tour bán chạy nhất, tour bán ít nhất để dễ dàng tùy chỉnh trong tương lai. Xem được tình trạng của các đơn hàng đang giao |
| Chức năng theo dõi các thành viên trong hệ thống | Admin | Admin có thể xem được các tài khoản đang có trong hệ thống, có thể là khách hàng, và các admin, nhân viên khác. |
| Chức năng theo dõi, quản lý đơn hàng | Nhân viên | Nhân viên có thể theo dõi, quản lý các đơn hàng sau khi khách hàng đặt hàng ( nhân viên có nghĩa vụ phải cập nhật trạng thái của đơn hàng và thông báo cho khách hàng nếu cần thiết) |
| Chức năng theo dõi các khách hàng có trong hệ thống | Nhân viên - Admin | Nhân viên có thể xem được các tài khoản khách hàng đang có trong hệ thống |

* Chức năng đăng nhập: phân quyền người dùng( khách hàng, nhân viên, admin).
* Khách hàng khi đăng nhập vào trang thì sẽ chuyển hướng sang trang chủ chứa các thông tin các tour
* Nhân viên khi đăng nhập thành công sẽ chuyển hướng trang quản lý dành cho nhân viên (quản lý tour, doanh thu, khách hàng)
* Admin khi đăng nhập vào trang ví dụ : /wp-admin thì sẽ chuyển hướng vào trang quản lý người dùng, quản lý các nhân viên

KHÁCH HÀNG

* Chức năng tạo đơn, xác nhận đơn, hoàn tất đơn. Chức năng đánh giá tour, gửi phản hồ sau khi hoàn tất đơn. Chức năng chỉnh sửa đơn hàng nếu đơn hàng chưa được thanh toán. Chức năng gửi email quảng cáo/khuyến mãi (chỉ với những người dùng đăng ký).

NHÂN VIÊN:

* + Nhân viên có thể thực hiên thêm mới, cập nhật, xóa các tour
  + Nhân viên có thể thống kê thu - chi/ số lượng đơn hàng.
  + Nhân viên có thể quản lý tình trạng các đơn hàng
  + Nhân viên có thể khách hàng

ADMIN ( NHÀ QUẢN LÝ )

* Theo dõi số thành viên trong hệ thống - có thể là các quản lý, các khách hàng, nhân viên (tên tài khoản, email, vai trò, …)
* Chức năng theo dõi thu-chi
* Theo dõi tình trạng đơn hàng đơn hàng
* Quản lý các tài khoản nhân viên

**Layout của giao diện:**

**Partiton**

**Phân chia dữ liệu – Partition**

1. **Partition:**

* *Partition (Phân chia dữ liệu):* là một giải pháp chia nhỏ một Database lớn thành nhiều Database nhỏ, ta có thể phân tách từng bảng hoặc cả một DB ra nhiều phần nhỏ đặt ở nhiều máy chủ (server) khác nhau. Điều này sẽ giúp cho hệ thống DB của chúng ta đạt được các tính chất khả năng bảo trì, hiệu xuất, tính sẵn sàng, và cân bằng tải của ứng dụng. Và giải pháp này cũng giảm chi phí cũng như tính mở rộng để scale up DB bằng cách dùng nhiều server nhỏ gộp lại hơn là nâng cấp một server lớn.
* Partition function: Ánh xạ các dòng của bảng hoặc các chỉ mục trong phân mảnh dựa theo tiêu chí phân mảnh. Qui định giá trị biên cho các đoạn. Hệ thống dựa vào hàm này để xác định đoạn mà mỗi bản ghi thuộc vào.
* Partition Scheme: Ánh xạ các phân mảnh của bảng vào các filegroup (mỗi filegroup chứa một phân mảnh)

1. **Lợi ích của việc sử dụng Partition:**

* Cải thiện khả năng co giãn và khả năng quản lý của các bảng lớn (large table) và các bảng có các kiểu truy cập khác nhau.
* Khi các bảng và các chỉ mục quá lớn, việc phân mảnh giúp chia dữ liệu thành cá phần nhỏ hơn, có thể quản lý được.
* Nếu 1 bảng lớn tồn tại trên hệ thống nhiều CPUs, việc phân mảnh bảng sẽ giúp tăng hiệu suất khi thực hiện song song
* *Việc backup/restore* một đoạn mà không ảnh hưởng đến các đoạn còn lại.

🡺 Phân chia dữ liệu sẽ giúp cho hệ thống DB đạt được các tính chất khả năng bảo trì, hiệu xuất, tính sẵn sàng, và cân bằng tải của ứng dụng. Và giải pháp này cũng giảm chi phí cũng như tính mở rộng để scale up DB bằng cách dùng nhiều server nhỏ gộp lại hơn là nâng cấp một server lớn.

1. **Các kiểu phân chia dữ liệu:**

* Phân chia theo chiều ngang (horizontal partition):
* Phân chia các dòng trong một table thành nhiều table khác nhau.
* Tình huống áp dụng: khi nhiều người dùng khác nhau cần truy cập các dòng dữ liệu khác nhau.
* Ưu điểm: tối ưu hoá tốc độ truy cập dữ liệu
* Nhược điểm:

Phức tạp khi phải truy cập toàn bộ dữ liệu.

Nếu không phân chia cẩn thận sẽ dễ gây mất cân bằng giữa các DB.

* Phân chia theo chiều dọc (vertical partition):
* Phân chia một cấu trúc luận lý thành những cấu trúc lưu trữ vật lý khác nhau. Phân chia dựa vào thuộc tính (cột) của bảng thành nhiều bảng khác nhau.
* Tình huống áp dụng: chia bảng thành các loại cấu trúc có giá trị sử dụng khác nhau hoặc cấu trúc truy cập thường xuyên và không thường xuyên.
* Ưu điểm: Dễ thực hiện và không ảnh hưởng đến ứng dụng
* Nhược điểm: Khi hệ thống lớn thì dữ liệu cũng lớn theo, do đó phải thực hiện phân chia nhiều lần.

1. **Phân chia dữ liệu bảng OrderTour (Hoá đơn) theo chiều ngang**

**Bước 1: Tạo các file group**

* Tạo các filegroup để chứa các phân mảnh sau khi đã phân chia.
* Group 1 (Primary): lưu các đơn hàng trước năm 2020 (ngày 31/12/2019 trở về trước)
* Group 2 (FG2): lưu các đơn hàng trong năm 2020 (từ ngày 1/1/2020 đến 31/12/2020)
* Group 3 (FG3): lưu các đơn hàng sau năm 2020 (từ ngày 1/1/2021 trở về sau)

**CREATE DATABASE** **DB\_OrderTour\_Partition**

ON PRIMARY

FILEGROUP FG1

(NAME= 'OrderTour\_Partition\_1', FILENAME='D:\PartitionDB\FG1\OrderTour\_Partition\_1.mdf', SIZE=2, MAXSIZE=100, FILEGROWTH=1 ),

FILEGROUP FG2

(NAME = 'OrderTour\_Partition \_2', FILENAME = 'D:\PartitionDB\FG2\OrderTour\_Partition\_2.ndf', SIZE = 2, MAXSIZE=100, FILEGROWTH=1 ),

FILEGROUP FG3

(NAME = 'OrderTour\_Partition\_3', FILENAME='D:\PartitionDB\FG3\OrderTour\_Partition\_3.ndf', SIZE = 2, MAXSIZE=100, FILEGROWTH=1 )

**GO**

**Bước 2: Tạo Partition Function**

* Tạo các giá trị biên cho mỗi bảng dựa vào cột createDate trong bảng chính (OrderTour)
* Hàm Partition function ‘OrderDatePartition’ tạo được 3 bảng nhỏ chia dữ liệu theo các mốc thời gian: trước ngày 1/1/2020; từ 1/1/2020 đến 31/12/2020; từ 1/1/2021 trở về sau.
* RANGE RIGHT nghĩa là lấy < or >=

LEFT lấy <= and >

|  |  |
| --- | --- |
| <1/1/2020 | OrderTour\_Partition\_1 |
| >=1/1/2020 and <31/12/2020 | OrderTour\_Partition\_2 |
| >=1/1/2021 | OrderTour\_Partition\_3 |

**Use** DB\_OrderTour\_Partition

**GO**

**CREATE PARTITION FUNCTION** OrderDatePartition (datetime)

AS

RANGE RIGHT

FOR VALUES ( '2020-01-01', '2021-01-01')

**GO**

**Bước 3: Tạo partition schema**

* Hàm Partition Scheme ‘OrderDatePartitionScheme’ dùng hàm ‘OrderDatePartition’ để “gắn” các đoạn vào từng filegroup

**Use** DB\_OrderTour\_Partition

**GO**

**CREATE PARTITION SCHEME** OrderDatePartitionScheme

AS

PARTITION OrderTourDatePartition

TO ([PRIMARY], FG2, FG3 )

**GO**

**Bước 4: Tạo partitioned table**

* Tạo một bảng mới từ Partition Scheme, bảng này sẽ lọc ra những dữ liệu cần tìm kiếm nhiều khi truy vấn gồm mã đơn, ngày lập hoá đơn và thành tiền.
* Vì ngày lập hoá đơn không phải khoá chính nên sẽ phải tạo thêm một index cho bảng mới theo cột ngày lập (date) để truy vấn được dễ dàng và nhanh chóng hơn.

**Use** DB\_Order\_Partition

**GO**

**CREATE TABLE** OrderTour\_Partition\_Table

(

IdOrder INT IDENTITY(1,1) primary key,

IdClient int,

createDate datetime default getdate(),

Total float)

**ON** OrderDatePartitionScheme (date);

**GO**

**CREATE INDEX** idx\_date\_order\_partition

**ON** Order\_Partition\_Table (date);

**GO**