**LỜI MỞ ĐẦU**

Trong bối cảnh công nghệ thông tin không ngừng phát triển, việc hiểu rõ và ứng dụng các kĩ thuật lập trình để giải quyết những vấn đề cụ thể là vô cùng cần thiết. Đặc biệt, trong lĩnh vực lập trình, các thư viện toán học đóng vai trò quan trọng, giúp tối ưu hóa quá trình xử lý và phân tích dữ liệu. Chính vì lẽ đó, môn học "Kĩ thuật lập trình" đã trang bị cho chúng ta những kiến thức nền tảng và kỹ năng thực hành quý báu để phát triển các ứng dụng và thư viện phần mềm hiệu quả.

Hướng tới mục tiêu áp dụng những kiến thức đã học vào thực tiễn, em chọn chủ đề "Xây dựng một thư viện toán học làm việc với số nguyên" cho bài báo cáo này. Thư viện toán học là một tập hợp các hàm và tiện ích giúp xử lý các phép toán cơ bản và nâng cao liên quan đến số nguyên, từ đó hỗ trợ lập trình viên trong việc triển khai các thuật toán một cách dễ dàng và hiệu quả hơn.

Qua một thời gian được học tập và thực hành liên quan tới những bài tập cơ bản về môn học, em xin được chọn chủ đề cho bài báo cáo cá nhân cuối kì là: “Xây dựng một thư viện toán học làm việc với số nguyên lớn”. Ở báo cáo này, em đã vận dụng các kiến thức được học và tìm hiểu thêm bên ngoài để áp dụng, xây dựng chương trình đơn giản để thực hiện các chức năng tính toán của số nguyên lớn

Em xin chân thành cảm ơn thầy Vũ Thành Nam, giảng viên chính lớp Kỹ thuật lập trình kỳ 2023.2 đã tận tình giảng dạy và hướng dẫn chúng em trong suốt học kì vừa qua. Em không chỉ được tiếp thu và nâng cao được kiến thức về môn học này mà còn được thầy trau dồi về những kỹ năng khác để vận dụng vào những dự án thực tế và xây dựng cách tự học làm sao cho hiệu quả. Tuy bài báo cáo đã được em chuẩn bị cẩn thận nhưng sẽ không thể tránh được những sai sót. Rất mong sau khi thực hiện, em sẽ có cơ hội được gặp lại và nghe thầy đưa ra những đóng góp, góp ý để bài báo cáo của em được hoàn thiện hơn.

Em xin chân thành cảm ơn !

Hà Nội, ngày 16 tháng 6 năm 2024

Sinh viên thực hiện

Lê Quang Đức

1. Mô tả cấu trúc chương trình
2. Giới thiệu về chương trình

Chương trình mà cá nhân em thực hiện là xây dựng một thư viện phép toán với số nguyên lớn theo các yêu cầu cụ thể sau:

* Số nguyên có độ lớn lên tới 1000 chữ số
* Nhận các số hạng từ bàn phím hoặc đọc từ file
* Thực hiện các phép tính toán: cộng, trừ, nhân, chia
* Hiển thị thông tin phép tính ( bao gồm kết quả), lưu ra file.

Và sau khi cân nhắc và dựa trên kiến thức mà em đã học được thì em đã thực hiện chương trình bằng ngôn ngữ lập trình C++.

Tất cả các yêu cầu của thầy đã được em thực hiện trong bài báo cáo. Ngoài ra em đã thực hiện thêm một số phần để hoàn hiện chương trình hơn

* Thêm vào chương trình các hàm dùng để so sánh ( <,>,=,>=.<= ) giữa hai số nguyên lớn
* Thêm vào chương trình các hàm dùng thực hiện tìm ước chung lớn nhất, bội chung nhỏ nhất, tìm trị tuyệt đối của số nguyên âm…
* Sử dụng thuật toán Karatsuba để nhân hai số nguyên lớn giúp giảm độ phức tạp tính toán học
* Thêm vào các hàm hỗ trợ in kết quả ra màn hình và ra file…

1. Cấu trúc của chương trình

Đầu tiên em đã phát triển chương trình của em có cấu trúc như sau:

BTL/:

├── main.cpp

│ ├ Struct ‘bigint’ # Định nghĩa cấu trúc ‘bigint’ và các phép toán

│ ├void /\* Các hàm thực hiện chức năng in kết quả \*/ ()

│ └Hàm main // Hàm chính để thực hiện chương trình

└── Kiem\_thu/

├── Ket\_Qua.txt //File để lưu kết quả

└── (Các trường hợp kiểm thử) .txt //File các trường hợp kiểm thử

A diagram of a company

Description automatically generated

Hình I.2: Mô hình thực thể liên kết ER

Sơ đồ trên minh họa việc quản lý quy trình giao hàng, thông tin chi tiết về các đơn hàng với các yêu tố như thông tin của đơn hàng, sản phẩm, các cửa hàng bán sản phẩm, các trung tâm thương mại,…

Các mối quan hệ chủ yếu là 1-n, trong đó có quan hệ giữa Orders và Orderdate là 1-1 do mỗi đơn hàng sẽ chỉ có một thông tin về ngày sản xuất và ngày vận chuyển duy nhất; quan hệ giữa Orders và Drivers là quan hệ N-N biểu thị rằng một đơn hàng có thể có nhiều người vận chuyển và một người vận chuyển có thể chở nhiều đơn hàng.

Cụ thể hơn sẽ được trình bày ở phần dưới đây.

1. **Chuyển đổi từ ER sang RE.**
   1. **Xác định thực thể**

Các tập thực thể chuyển thành các quan hệ có cùng tên và tập thuộc tính. Dựa trên sơ đồ hình trên ta có thể xác định các quan hệ sau

* Orders (order\_id, status, fee, amount, cost)
* payments (payments\_id, amount, fee, method, status)
* channels (channel\_id, type, name)
* storewallet (account\_id, account\_amount, order\_account)
* orderdate (order\_id, moment\_created, moment\_accepted, moment\_delivering,

moment\_finished)

* hubs (hubs\_id, name, city, state, latitude, longitude)
* drivers (driver\_id, modal, status)
* products (product\_id, category, unit\_price)

Trong đó, các thuộc tính được gạch chân ( ví dụ thuộc tính order\_id trong thực thể Orders) là khóa chính của thực thể đó.

* 1. Xác định các mỗi liên kết.

1. Mối liên kết 1-1:

Quan hệ này sẽ chứa khóa chính của quan hệ kia và ngược lại

A diagram of a diagram

Description automatically generated

Hình I.3.2.a. Mối liên kết 1 – 1

Trên hình là mô tả quan hệ tương ứng liên kết 1 – 1 giữa Orders và Orderdate.

Liên kết Dated: chúng ta bổ sung thuộc tính khóa chính của quan hệ này vào quan hệ kia và ngược lại để thể hiện liên kết Dated, chỉ ra mỗi mã đơn hàng sẽ chỉ ứng với thông tin của một ngày vận chuyển duy nhất.

Do hai quan hệ Orders và Orderdate có cùng khóa chính là order\_id nên khi ta tham chiếu thuộc tính vào sẽ không thay đổi.

- orders (order\_id, status, fee, amount, cost)  
- orderdate (order\_id, moment\_created, moment\_accepted, moment\_delivering, moment\_finished).

b. Mối liên kết 1 - N

Quan hệ đầu nhiều sẽ chứa khóa chính của quan hệ đầu một

A diagram of payment payments

Description automatically generated

Hình I.3.2.b: Mối liên kết 1 – N

Trên hình là ví dụ mô tả quan hệ tương ứng liên kết 1 – N giữa Orders – Payments và   
Orders – Channels

Liên kết Recorded: chúng ta bổ sung thuộc tính Order\_id vào Payments để thể hiện liên kết Recorded, chỉ ra mỗi đơn hàng có thể được ghi trong nhiều hóa đơn, khi đó:  
payments( payment\_id, amount, status, method, fee, order\_id), với order\_id là khóa ngoài của payments tham chiếu đến Orders.

Tương tự với các liên kết 1 – N khác ta được :

* Orders (order\_id, store\_id, channel\_id,product\_id, status, fee, amount, cost)
* channels (channel\_id, type, name)
* store\_wallet (account\_id, store\_id, account\_amount, order\_account)
* orderdate (order\_id, moment\_created, moment\_accepted, moment\_delivering, moment\_finished)
* hubs (hubs\_id, name, city, state, latitude, longitude)
* drivers (driver\_id, modal, status)
* stores (store\_id, hub\_id, name, segment, plan\_price, latitude, longtitude)
* products (product\_id, category, unit\_price)

c. Mối liên kết N – N.

Sẽ có thêm quan hệ mới với các thuộc tính là khóa chính ở các thực thể và các thuộc tính riêng của nó

A diagram of a diagram

Description automatically generated

Hình I.3.2.c: Mối liên kết N – N

Trên hình là ví dụ mô tả quan hệ tương ứng liên kết N – N giữa Orders – Drivers.

Liên kết Deliverd: chúng ta tạo một lược đồ quan hệ mới Deliveries( order\_id, driver\_id, meter, status, delivery\_id ) với delivery\_id là khóa chính, {order\_id, driver\_id} là khóa ngoại trong đó order\_id tham chiếu đến Orders và driver\_id tham chiết drivers, {status, meters} chỉ ra quãng đường vận chuyển và trạng thái giao hàng.

Sau khi xác định thực thể và mối liên kết thực thể, ta có cơ sở dữ liệu quan hệ hoàn chỉnh sau:

* Orders (order\_id, store\_id, channel\_id,product\_id, status, fee, amount, cost)
* payments (payments\_id, order\_id, amount, fee, method, status)
* channels (channel\_id, type, name)
* store\_wallet (account\_id, store\_id, account\_amount, order\_account)
* orderdate (order\_id, moment\_created, moment\_accepted, moment\_delivering, moment\_finished)
* hubs (hubs\_id, name, city, state, latitude, longitude)
* drivers (driver\_id, modal, status)
* stores (store\_id, hub\_id, name, segment, plan\_price, latitude, longtitude)
* deliveries (delivery\_id, order\_id, driver\_id, meters, status)
  1. Mô hình cơ sở dữ liệu quan hệ

A diagram of a computer

Description automatically generated with medium confidence

Hình 3.3: RE diagram

Chú thích:

: Quan hệ 1 – 1   
 : Quan hệ 1 – N   
 : Khóa chính  
 : Khóa noại

1. Thông tin về cơ sở dữ liệu
2. Tổng quan về cơ sở dữ liệu   
   Tên cơ sở dữ liệu: Delivery Center: Food & Goods orders in Brazil ( nguồn Kaggle) .

Kích thước cơ sở dữ liệu : 162 MB  
Số lượng bản ghi: 10 bảng

A screenshot of a computer

Description automatically generated

Hình II.1.1: Tổng quan CSDL

Dữ liệu mô phỏng lại hệ thống quản lí các đơn hàng, là nền tảng tích hợp các nhà bán lẻ và chợ với gần 900000 đơn hàng, tạo ra một hệ sinh thái lành mạnh để bán hàng hóa và thực phẩm trong ngành bán lẻ Brazil.

A screenshot of a computer

Description automatically generated

Hình II.1.2: Thông tin từng bảng của cơ sở dữ liệu

1. Chi tiết các bảng
   1. Bảng channels.

Mô tả thông tin về kênh quảng cáo sản phẩm

A screenshot of a computer

Description automatically generated

Hình II.2.1: Bảng Channels

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| channel\_id | Chính | INT | Số hiệu của kênh |
| channel\_name | X | VARCHAR(45) | Tên của kênh bán |
| channel\_type | X | VARCHAR(45) | Thể loại kênh |

Dựa vào bảng trên ta có thể xác định được những từng kênh xem có quảng cáo những mặt hàng hàng hóa nào.

* 1. Bảng Deliveries

Mô tả thông tin về các chuyến vận chuyển

A screenshot of a data

Description automatically generated

Hình II.2.2: Bảng Deliveries

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| delivery\_id | chính | INT | Mã số vận chuyển |
| delivery\_order\_id | ngoại | BIGINT | Mã đơn hàng |
| driver\_id | ngoại | INT | Mã người vận chuyển |
| distance\_meters | X | INT | Quảng đường vận chuyển |
| status | X | VARCHAR(45) | Trạng thái vận chuyển |

Dựa vào bảng trên ta có thể xác định được thông tin về chuyến vận chuyển các đơn hàng về quãng đường vận chuyển và trạng thái giao hàng.

* 1. Bảng Drivers

A screenshot of a computer

Description automatically generated

Hình II.2.3: Bảng Drivers

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| driver\_id | chính | INT | Mã số người giao hàng |
| driver\_modal | X | VARCHAR(45) | Phương tiện giao hàng |
| driver\_type | X | VARCHAR(45) | Phương thức vận chuyển |

Dựa vào bảng trên ta có thể xác định được thông tin về người giao hàng giúp cho chúng ta biết được người giao hàng đó sử dụng phương tiên gì và phương thức giao hàng.

* 1. Bảng Hubs

A screenshot of a table

Description automatically generated

Hình II.2.4: Bảng Hubs

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| hub\_id | chính | INT | Mã trung tâm thương mại |
| hub\_name | X | VARCHAR(45) | Tên trung tâm thương mại |
| hub\_city | X | VARCHAR(45) | Trung tâm ở thành phố nào |
| hub\_state | X | VARCHAR(45) | Trung tâm ở quận nào |
| hub\_latitude | X | DOUBLE | Vĩ độ của trung tâm |
| hub\_longtitude | X | DOUBLE | Kinh độ của trung tâm |

Dựa vào bảng trên ta có thể xác định được thông tin về các trung tâm thương mại, giúp chúng ta xác định được vị trí của trung tâm thương mại.

* 1. Bảng Orders.

A screenshot of a computer

Description automatically generated

Hình II.2.5: Bảng Orders

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| order\_id | chính | BIGINT | Mã đơn hàng |
| store\_id | ngoại | INT | Mã cửa hàng |
| channel\_id | ngoại | INT | Số hiệu kênh bán sản phẩm |
| product\_id | ngoại | INT | Mã sản phẩm |
| order\_status | X | VARCHAR(45) | Trạng thái đơn hàng |
| order\_amount | X | DOUBLE | Lượng đơn hàng |
| order\_delivery\_fee | X | DOUBLE | Phí vận chuyển |
| order\_delivery\_cost | X | DOUBLE | Giá vận chuyển |

Dựa vào bảng trên ta có thể xác định được thông tin về các đơn hàng, giúp chúng ta biết được mỗi đơn hàng sẽ được bày bán ở cửa hàng nào, được quảng cáo ở đâu và cho chúng ta biết thông tin vận chuyển đơn hàng.

* 1. Bảng OrderDate:

A screenshot of a computer

Description automatically generated

Hình II.2.6: Bảng OrderDate

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| order\_id | chính | BIGINT | Mã đơn hàng |
| moment\_created | X | DATE | Ngày tạo đơn hàng |
| moment\_accepted | X | DATE | Ngày đơn hàng được chấp nhận |
| moment\_delivering | X | DATE | Ngày đơn hàng vận chuyển |
| moment\_finish | X | DATE | Ngày đơn hàng hoàn thành |

Dựa vào bảng trên ta có thể xác định được thông tin về ngày tạo đơn hàng, ngày giao hàng và ngày hoàn thành các đơn hàng

* 1. Bảng Payments

A screenshot of a computer

Description automatically generated

Hình II.2.7: Bảng Payments

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| payment\_id | chính | BIGINT | Mã hóa đơn |
| payment\_order\_id | ngoại | BIGINT | Mã đơn hàng |
| payment\_amount | X | DOUBLE | Giá trị của đơn hàng |
| payment\_fee | X | DOUBLE | Thuế |
| payment\_method | X | VARCHAR(45) | Phương thức thanh toán |
| payment\_status | X | VARCHAR(45) | Trạng thái hóa đơn |

Dựa vào bảng trên ta có thể xác định được thông tin về các tờ hóa đơn, giúp cho chúng ta biết thông tin đơn hàng được ghi trong hóa đơn nào, bao gồm giá thành, phương thức thanh toán và thuế của đơn hàng.

* 1. Bảng Stores

A screenshot of a computer

Description automatically generated

Hình II.2.8: Bảng Stores

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| store\_id | chính | INT | Mã cửa hàng |
| hub\_id | ngoại | INT | Mã trung tâm thương mại |
| store\_segment | X | VARCHAR(45) | Mặt hàng kinh doanh |
| store\_plan\_price | X | VARCHAR(45) | Giá dự bán |
| store\_latitude | X | DOUBLE | Vĩ độ của cửa hàng |
| store\_longitude | X | DOUBLE | Kinh độ của cửa hàng |

Dựa vào bảng trên ta có thể xác định được thông tin về các cửa hàng, số hiệu của các cửa hàng, mặt hàng mà cửa hàng kinh doanh và vị trí tọa độ của cửa hàng.

* 1. Bảng StoreWallet

A screenshot of a table

Description automatically generated

Hình II.2.8: Bảng StoreWallet

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| account\_id | chính | INT | Mã tài khoản |
| store\_id | ngoại | INT | Mã cửa hàng |
| account\_amount | X | INT | Ngân sách của tài khoản |
| order\_account | X | INT | Tài khoản của đơn hàng |

Dựa vào bảng trên ta có thể xác định được thông tin về ngân sách của các của hàng, cho biết được ngân sách tài khoản cửa mỗi ngân hàng là bao nhiêu.

* 1. Bảng Products

\A screenshot of a computer

Description automatically generated

Hình II.2.10: Bảng Products

Các thuộc tính

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Khóa** | **Kiểu dữ liệu** | **Mô tả** |
| product\_id | chính | INT | Mã sản phẩm |
| product\_name | X | VARCHAR(45) | Tên sản phẩm |
| category | X | VARCHAR(45) | Loại sản phẩm |
| unit\_price | X | DOUBLE | Giá của sản phẩm |

Dựa vào bảng trên ta có thể xác định được thông tin về các loại sản phẩm kinh doanh, bao gồm các thông tin về tên, giá cả và loại sản phẩm buôn bán.

1. Chuẩn hóa dữ liệu
   1. Chuẩn loại 1:

* Tất cả các ô đều chỉ chứa duy nhất một giá trị ( đơn trị )
* Mỗi bản ghi trong các bảng đều là duy nhất
* Tất cả các bảng đều đạt chuẩn loại 1 ( 1NF)

VD: Bảng Orders

A screenshot of a computer

Description automatically generated

Hình II.3.1. Bảng Orders

* 1. Chuẩn loại 2:
* Đạt chuẩn 1NF
* Tất cả các thuộc tính không khóa phụ thuộc hoàn toàn vào tập các thuộc tính khóa chính

Dưới đây là các bảng đạt chuẩn loại 2

A screenshot of a computer

Description automatically generated  
Nhận thấy rằng các bảng trên đều chỉ có một khóa duy nhất, nên các thuộc tính còn lại sẽ phụ thuộc hoàn toàn vào khóa chính

* Tất cả các bảng đều đạt chuẩn loại 2
  1. Chuẩn loại 3.
* Đạt chuẩn 2NF
* Các thuộc tính không khóa phải phụ thuộc trực tiếp vào khóa chính
* Tất cả các bảng đều đạt chuẩn 3 ngoại trừ bảng Hubs

A screenshot of a phone

Description automatically generated

Ta nhận thấy rằng từ thuộc tính ‘hub\_city’ có thể suy ra được thuộc tính ‘hub\_state’ và từ thuộc tính ‘hub\_latitude’ ‘hub\_longitude’ có thể suy ra được cả hai thuộc tính ‘hub\_city’ và ‘hub\_state’. Do vậy, bảng Hubs không đạt chuẩn 3NF.

Ví dụ về bảng đạt chuẩn 3NF:

A screenshot of a computer

Description automatically generated Để bảng Hubs đạt chuẩn 3NF, ta cải tiến như sau:

A screenshot of a computer

Description automatically generated

Với cách trên thì bảng Hubs đã đạt chuẩn 3NF

* 1. Chuẩn Boyce-Codd (BCNF)
* Đạt chuẩn 3NF
* Không có thuộc tính khoá mà phụ thuộc hàm vào thuộc tính không khoá.

A screenshot of a computer

Description automatically generated

Như đã nêu ở trên, đây là các bảng đạt chuẩn 3. Đồng thời các bảng trên không có bảng nào mà có thuộc tính khóa mà phụ thuộc hàm vào thuộc tính không khóa.

**TRUY VẤN DỮ LIỆU**

1. Các câu lệnh truy vấn

Kịch bản nghiệp vụ:

Cuối tháng, cấp trên yêu cầu tạo một báo cáo về tình hình công ty trong tháng vừa qua, sử dụng My SQL để truy vấn về các thông tin cần thiết trong CSDL

1. Thống kê các cửa hàng mới trong tháng của công ty ở các trung tâm mới, thể hiện mức độ mở rộng thị trường.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tự động tạo

Kết quả truy vấn:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tự động tạo

1. TỐI ƯU HÓA TRUY VẤN
2. Tìm hiểu về EXPLAIN

* Explain là câu lệnh được sử dụng để thu được kế hoạch thực thi truy vấn.
* Cách sử dụng : thêm lệnh EXPLAIN vào đầu mỗi query.

A screenshot of a computer

Description automatically generated

* Nó hoạt động với các mệnh đề SELECT, DELETE, INSERT, REPLACE, và UPDATE, và nó hiển thị thông tin từ trình tối ưu hóa về kế hoạch thực thi câu truy vấn.



* Các trường trong EXPLAIN
* Id : Là số thứ tự cho mỗi câu SELECT trong truy vấn của bạn (trường hợp bạn sử dụng các truy vấn lồng nhau - nested subqueries).
* Select\_type : Là tham số về kiểu query. Nó bao gồm các giá trị sau: simple, union …
* Table : Nó chỉ là tên bảng liên quan đến câu truy vấn.
* Type : Cách MySQL joins các bảng. Nó có thể chỉ ra các index bị thiếu và nó cũng có thể cho thấy câu truy vấn của bạn cần phải xem xét lại.
* Possible\_key : List tất cả các key bởi MySql để tìm ra các dòng trong bảng. Các key này có thể có hoặc không được sử dụng trong thực tế.
* Key : Key được chính thức MySql sử dụng để làm index để tìm kiếm. Cột này

có thể chứa khóa không được liệt kê ở cột possible\_keys.

* Key\_len : Hiển thị độ dài của index trình tối ưu hóa truy vấn chọn để sử dụng. Ví dụ, key\_len = 2 tức là cần bộ nhớ để lưu 2 ký tự.
* Ref : Hiển thị các cột hoặc các hằng số được so sánh với index được nêu ra ở

cột key. Trong trường hợp query là JOIN thì đây chính là giá trị của key ở bảng tương ứng mà được join cùng với bảng chính.

* Rows : Nó thể hiện số dòng mà mysql "dự định" sẽ fetch ra từ bảng trong

query đó. Đây là một chỉ số rất quan trọng, nhất là khi bạn dùng JOIN hoặc truy vấn con.

* Filtered : Cho biết phần trăm ước lượng của số hàng trong kết quả cuối cùng sau khi áp dụng các điều kiện lọc của truy vấn. Giá trị trong cột này thường dao động từ 0 đến 100.
* Extra : Các giá trị kiểu như Using,Temporary, Using filesort,... của cột này có thể cho thấy một truy vấn không thực sự tốt. Chỉ cần nhìn qua extra thì bạn sẽ đoán được phần nào chuyện gì sẽ xảy ra đằng sau một query nào đó.

**Kết luận:**

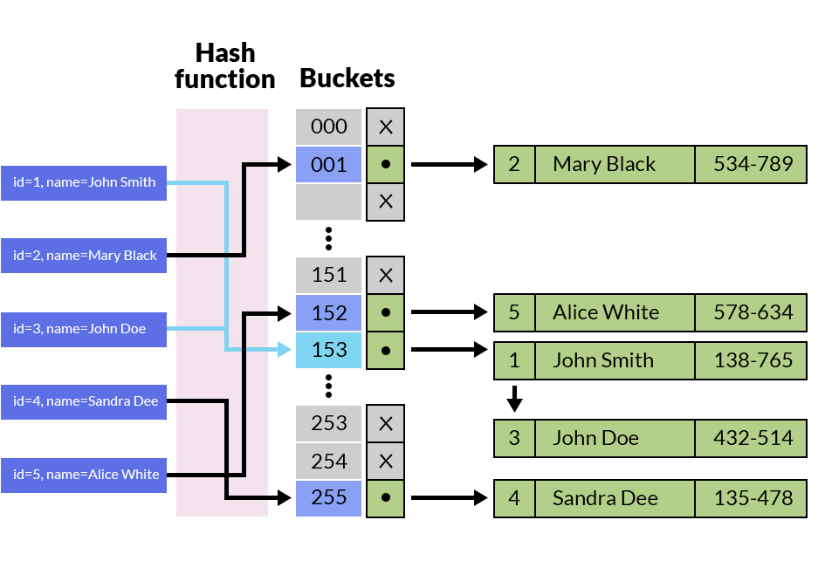
* Thông qua giá trị của id và select\_type, ta có thể biết được trình tự Access vào các bảng để lấy ra cái gì, kết hợp với bảng khác như nào.
* Thông qua giá trị của type, key, ref và rows, ta có thể biết ứng với mỗi bảng sẽ có những thông tin gì được fetch ra, truy cập vào bảng nào sẽ nặng, để chúng ta có thể đánh index cho bảng đó.
* Thông qua giá trị của extra, chúng ta sẽ có một cái nhìn tổng quát về query nào đó.

1. Các cách tối ưu truy vấn

* Sử dụng index
* Tối ưu hóa câu lệnh truy vấn
* Tối ưu qua kiểu dữ liệu
* Sử dụng partition

1. Sử dụng index

* Khái niệm :
* Chỉ mục (Index) là một cấu trúc dữ liệu có thể giúp xác định nhanh chóng các bản ghi trong bảng.
* Thực tế, Index cũng là loại bảng gồm hai cột, mà giữ khóa chính hoặc trường Index chứa các bản sao của bản ghi nó trỏ tới và một con trỏ trỏ đến từng bản ghi của bảng thực tế.
* Cấu trúc của một index:



* Các câu lệnh thực hiện khi sử dụng index :
* CREATE INDEX index\_name ON tbl\_name (column1,column2,..)
* ALTER TABLE tbl\_name
* ADD INDEX index\_name (column 1, column2,..)
* DROP INDEX index\_name;
* A close-up of words

  Description automatically generated SHOW INDEX FROM tbl\_name
* Khi sử dụng index trong CSDL :



* Ta thấy khi chưa sử dụng index, các cột possible\_keys, key, key\_len đều là Null. Đặc biệt, số dòng phải duyệt trong cột rows là 361809 rất lớn. Bây giờ ta tạo index cho cột sử dụng where:

A close-up of a text

Description automatically generated



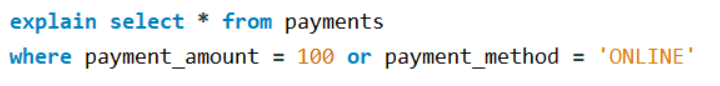
* Ta thấy sau khi tạo index, số dòng phải duyệt qua giảm xuống chỉ còn 14, giảm đi gấp 20.000 lần khi chưa tạo index – một con số đáng kinh ngạc.

1. Tối ưu hóa câu lệnh truy vấn

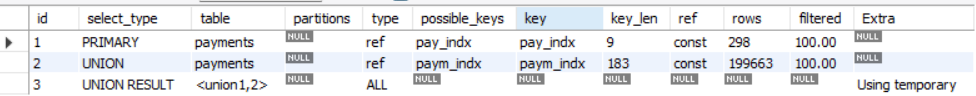
* Một số cách tối ưu câu lệnh truy vấn :
* Liệt kê các trường cần thiết thay vì select \*
* Hạn chế (**limit**) số lượng kết quả trả về
* Sử dụng **join** thay vì **sub query**
* Sử dụng **union** thay vì **or**
* Hạn chế sử dụng hàm ở vế phải toán tử
* Kĩ thuật CTE(bảng tạm) với **with** để thay **having** bằng **where**

Cách thứ nhất và thứ hai rất cơ bản, nó chỉ là giảm số lượng kết quả bản ghi trả về, chỉ trả về số lượng bản ghi cần thiết, vì vậy sẽ không trình bày chi tiết trong phần này.

* Sử dụng **union** thay vì **or :**





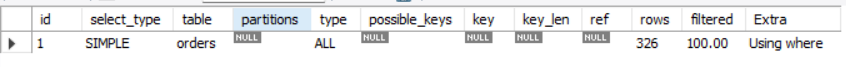
A close-up of a number

Description automatically generated

* Ta thấy sự khác biệt giữa hai câu lệnh truy vấn trên, khi sử dụng ‘or’, việc thực thi của Mysql chỉ lấy một key của một trường, trong khi đó cả hai trường ta đang sử dụng để truy vấn đều có khóa, việc sử dụng ‘or’ là không tối ưu, thay vào đó ta nên sử dụng ‘union’ kết hợp hai câu lệnh truy vấn để được kết quả như mong muốn và tối ưu hóa thời gian.

Ta thấy là khi sử dụng ‘or’ phải duyệt qua 399326 dòng, nhiều hơn gấp đôi khi ta sử dụng ‘union’ với hai câu lệnh. Tối ưu hơn gấp hai lần.

* Hạn chế sử dụng **hàm** ở vế trái toán tử :

A close-up of a number

Description automatically generated

* Ta thấy rằng việc sử dụng hàm ở vế trái làm tăng số dòng phải duyệt qua, số dòng lúc này là 326, trong khi không sử dụng và thay thế bằng câu truy vấn đơn giản khác, số dòng phải duyệt qua chỉ còn là 5. Giảm đi khoảng 60 lần.
* Kĩ thuật CTE(bảng tạm) với **with** để thay **having** bằng **where :**
* A white background with black text

  Description automatically generatedKhi ta sử dụng having :
* Khi ta sử dụng bảng with và dùng where :





* Ta thấy rằng, khi sử dụng having, thời gian truy vấn là 0.047 nhiều hơn gấp rưỡi khi sử dụng bảng tạm và dùng where. Ngoài ra, bảng tạm còn được sử dụng để select nhiều lần dữ liệu trong bảng mà không phải duyệt trên cả bảng chính.
* Sử dụng join thay vì sub query :
* Khi ta sử dụng sub query :

A close-up of a number

Description automatically generated



* Khi ta sử dụng join :

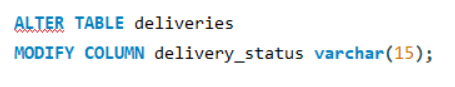
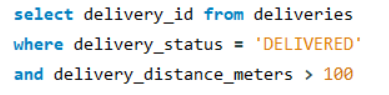
A close up of a website

Description automatically generated

* Khi ta sử dụng join, thời gian truy vấn chỉ còn 0.01 thay vì 0.016 như khi sử dụng sub query, thời gian giảm tới 16 lần.

1. Tối ưu hóa kiểu dữ liệu :

* Ta thấy cột delivery\_status chỉ có các dữ liệu : DELIVERIED, CANCELLED. Không cần thiết phải để kiểu dữ liệu là varchar(50). Vì vậy, ta đổi sang varchar(15) để tối ưu hơn. Ta thấy rằng, thời gian từ 0.047 giảm xuống gấp rưỡi khi ta thay đổi kiểu dữ liệu ngắn gọn và phù hợp hơn.



1. Sử dụng partition để tối ưu

* Partitioning theo đúng như tên của nó là việc phân chia một table thành những

phần nhỏ theo một logic nhất định, được phân biệt bằng key, key này thường là

tên column trong table.

* Điều đó cũng có nghĩa khi bạn tìm kiếm dữ liệu, database chỉ cần search trong

phạm vi một hoặc vài partition nào đó, giúp giảm thời gian truy vấn dữ liệu.

* Ưu điểm :
* Khả năng mở rộng
* Cải thiện hiệu suất trong cơ sở dữ liệu
* Tạo điều kiện thuận lợi cho việc quản lí, vận hành và bảo trì dữ liệu
* Các loại partition
* RANGE Partitioning
* HASH Partitioning
* LIST Partitioning
* ….
* **Range Partitioning :**

Phân vùng theo dải (range partitioning) là cách phân vùng dữ liệu dựa trên các phạm vi giá trị của một hoặc nhiều cột. Mỗi phân vùng chứa một dải giá trị khác nhau.

**Đặc điểm:**

* **Dễ hiểu và dễ quản lý**: Các dải giá trị cụ thể được xác định rõ ràng, giúp dễ dàng quản lý và duy trì.
* **Thích hợp cho dữ liệu tuần tự**: Rất hữu ích cho các cột có giá trị tuần tự hoặc có thứ tự tự nhiên, chẳng hạn như ngày tháng, số thứ tự, hay các mã định danh liên tiếp.
* **Truy vấn tối ưu**: Truy vấn trên một phạm vi giá trị cụ thể sẽ chỉ cần truy cập phân vùng chứa phạm vi đó, giảm đáng kể số lượng dữ liệu cần xử lý.

Khi trước khi tối ưu bằng phân vùng :





Khi tối ưu bằng phân vùng :

A white background with blue and orange text

Description automatically generated



* Ta thấy sau khi được phân vùng, việc tìm kiếm chỉ diễn ra trong phân vùng, giảm tối đa lượng dòng cần duyệt từ hơn 350000 dòng xuống chỉ còn một dòng, gấp 350000 lần.
* **Hash Partitioning :**

Phân vùng theo băm (hash partitioning) là cách phân vùng dữ liệu dựa trên hàm băm của một hoặc nhiều cột. Dữ liệu được phân phối ngẫu nhiên vào các phân vùng dựa trên kết quả của hàm băm.

**Đặc điểm:**

* **Phân phối đồng đều**: Giúp phân phối dữ liệu đều hơn giữa các phân vùng, tránh tình trạng một phân vùng chứa quá nhiều dữ liệu so với các phân vùng khác (skew).
* **Không tuần tự**: Không phụ thuộc vào thứ tự giá trị của cột, mà dựa vào giá trị băm của cột đó.
* **Khó quản lý và mở rộng**: Khó xác định cụ thể phân vùng chứa một giá trị nào đó và khó mở rộng thêm phân vùng khi cần thiết.

A close up of numbers

Description automatically generatedTrước khi được tối ưu bằng phân vùng :



Sau khi được tối ưu :

A close-up of a sign

Description automatically generated



* Ta thấy được, sau khi được phân vùng, số dòng được duyệt được giảm đi từ 400000 dòng xuống chỉ còn hơn 13 nghìn dòng, gấp gần hơn 40 lần.
* Từ đó, ta rút ra đánh giá giữa hai loại parttition kể trên :

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **Range Partition** | **Hash Partition** |
| **Thời gian** | Phân vùng dựa trên các giá trị phạm vi, thường dễ xác định | Phân vùng dựa trên hàm băm, phụ thuộc vào độ phức tạp của hàm băm |
| **Tốc độ** | Tìm kiếm trong phạm vi cụ thể, thường nhanh hơn khi dữ liệu được phân phối đều | Tìm kiếm phụ thuộc vào kết quả hàm băm, tốc độ có thể không đồng đều |
| **Ưu điểm** | - Dễ dàng quản lý và hiểu  - Tốt cho dữ liệu có phạm vi rõ ràng  - Tối ưu cho truy vấn theo phạm vi | - Phân phối dữ liệu đồng đều hơn  - Tránh được điểm nóng dữ liệu  - Tối ưu cho truy vấn với khóa duy nhất |
| **Nhược điểm** | - Có thể dẫn đến phân phối dữ liệu không đều  - Điểm nóng dữ liệu nếu phạm vi không được chọn đúng | - Phức tạp hơn để quản lý  - Khó dự đoán phân phối dữ liệu  - Tối ưu cho các truy vấn cụ thể có thể khó khăn |
| **Khi nào nên dùng** | - Dữ liệu có phạm vi rõ ràng, như ngày tháng, số lượng  - Truy vấn thường xuyên dựa trên các phạm vi xác định  - Muốn dễ dàng quản lý và hiểu cấu trúc phân vùng | - Dữ liệu không có phạm vi rõ ràng  - Muốn tránh điểm nóng dữ liệu  - Truy vấn chủ yếu dựa trên khóa duy nhất hoặc hàm băm |

1. Tổng kết

**Các cách tối ưu truy vấn :**

1. Sử dụng index
2. Tối ưu hóa câu lệnh truy vấn

* Liệt kê các trường cần thiết thay vì select \*
* Hạn chế (**limit**) số lượng kết quả trả về
* Sử dụng **join** thay vì **sub query**
* Sử dụng **union** thay vì **or**
* Hạn chế sử dụng hàm ở vế phải toán tử
* Kĩ thuật CTE(bảng tạm) với **with** để thay **having** bằng **where**

1. Tối ưu qua kiểu dữ liệu
2. Sử dụng partition

* RANGE PARTITION
* HASH PARTITION

**CẬP NHẬT DỮ LIỆU**

1. PROCEDURE

Procedure (Thủ tục) là một chương trình trong cơ sở dữ liệu gồm nhiều câu lệnh

mà bạn lưu lại cho những lần sử dụng sau. Trong MYSQL, bạn có thể truyền các

tham số vào Procedure, tuy nó không trả về một giá trị cụ thể như Function (hàm) nhưng

cho biết việc thực thi thành công hay thất bại.

Có 3 loại Procedure cơ bản: Procedure Insert, Procedure Update, Procedure Delete. Trong báo cáo này, chúng em thực hiện tổng cộng 27 procedures insert, update, delete dữ liệu trong từng bảng.

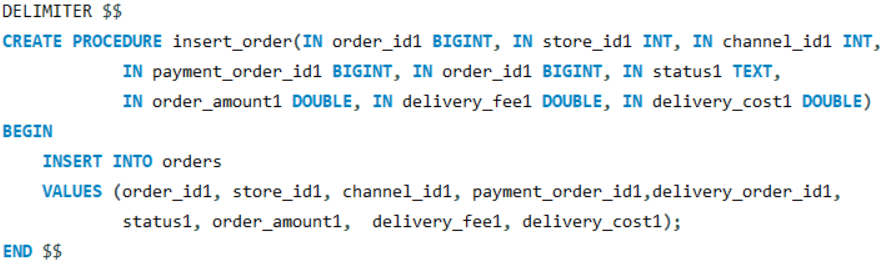
Ngoài ra còn có

* 5 procedure nghiệp vụ, nâng cao
* 5 transaction nghiệp vụ thực tế

Sử dụng con trỏ, biến cục bộ, vòng lặp, if else, prepare statement, tạo event, case when, biến người dùng và một số cấu trúc câu lệnh khác.

Vì nội dung còn hạn chế nên nhóm em xin phép trình bày 3 procedure của 3 tác vụ cơ bản được sử dụng cho bảng ‘orders’ và các procedure nâng cao.

1. PROCEDURE INSERT

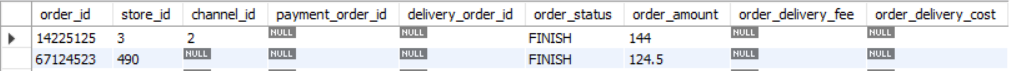


Trong procedure trên, tham số truyền vào là các trường của bảng ‘orders’, câu lệnh trong procedure là câu lệnh INSERT INTO.

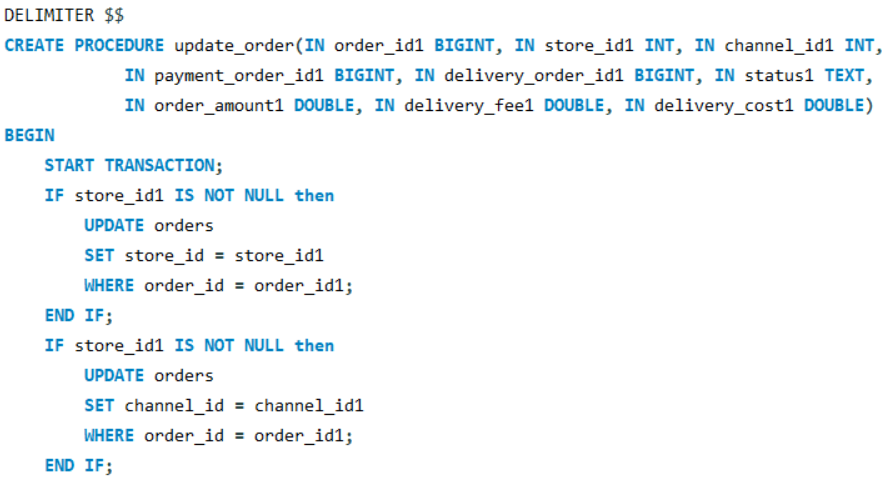
Khi thực hiện, ta sử dụng lệnh CALL :



Và đây là kết quả sau khi thực thi, dữ liệu đã được thêm vào bảng thành công :



1. PROCEDURE UPDATE



A screen shot of a computer code

Description automatically generated

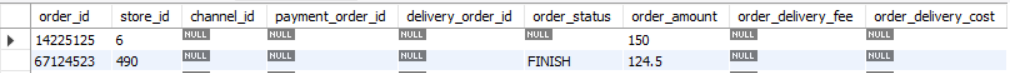
A screenshot of a computer code

Description automatically generated

Khi Khi thực hiện, ta sử dụng lệnh CALL :



Và đây là kết quả sau khi thực thi, dữ liệu đã được sửa thành công :



1. PROCEDURE DELETE

A screen shot of a computer code

Description automatically generated

A computer screen shot of a code

Description automatically generated

Các điểm lưu ý :

* Sử dụng câu lệnh hệ thống để lấy tên bảng tham chiếu, tên trường tham chiếu, tên bảng được tham chiếu và tên trường được tham chiếu

A close-up of a computer code

Description automatically generated

* Sử dụng con trỏ để trỏ đến câu lệnh **select** trên
* Sử dụng vòng lặp while để xóa dữ liệu trong các trường của các bảng được tham chiếu trước, cuối cùng mới xóa đến dữ liệu trong bảng chính

Khi thực hiện, ta sử dụng lệnh CALL :

Và đây là kết quả sau khi thực thi, dữ liệu đã được xóa thành công khỏi các bảng :





1. PROCEDURE NGHIỆP VỤ THỰC TẾ

* Bài toán 1 : Đưa ra số lượng đơn hàng của từng store trong tháng của năm nào đó

A white background with black and blue text

Description automatically generated

 A screenshot of a computer

Description automatically generated

* Bài toán 2 : Tổng doanh thu của đơn vị vận chuyển trong tháng và đưa ra sự so sánh so với tháng trước

A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

Một số điểm đặc biệt :

* Sử dụng window function LAG
* CASE WHEN, biến cục bộ và subquery

Khi thực hiện lệnh gọi :

A screenshot of a computer

Description automatically generated

* Bài toán 3 : Lấy ra top 5 stores có giá trị đơn hàng cao nhất, thấp nhất trong tháng nào đó

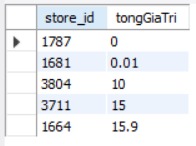
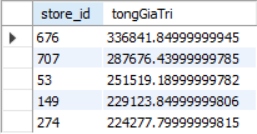
Một số điểm đặc biệt :

* Sử dụng IFELSE
* Các loại join

A screenshot of a computer code

Description automatically generated

Khi thực hiện lệnh chạy :



* Bài toán 4 : Sau một thời gian, các đơn hàng bị hủy không còn giá trị sử dụng, chúng được tự động xóa khỏi bảng orders.

Một số điểm đặc biệt :

* Sử dụng con trỏ, biến cục bộ
* Sử dụng câu lệnh tạo sự kiện tự động chạy sau một thời gian chỉ định

A screen shot of a computer code

Description automatically generatedA screen shot of a computer code

Description automatically generated

A computer code with text

Description automatically generated with medium confidence

* Bài toán 5 : Mỗi 6 tháng công ty lấy ra 1 store có tổng số tiền đặt hàng lớn nhất nhiều nhất sẽ được nhận triết khấu, đãi ngộ, ...

A computer screen shot of a code

Description automatically generated

A computer screen shot of a program

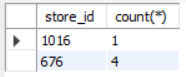
Description automatically generated

Một số điểm đặc biệt :

* Sử dụng 3 window function
* Kết hợp sử dụng with tạo bảng tạm lấy thông tin 6 tháng liên tiếp

Khi thực hiện lệnh chạy :



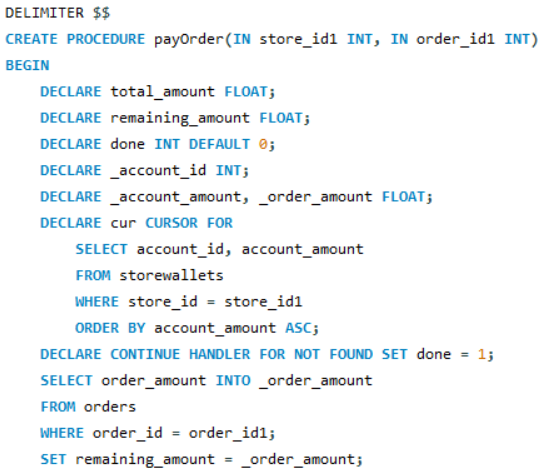


1. TRANSACTION
2. Định nghĩa

* Transaction trong SQL là một nhóm các câu lệnh SQL.
* Nếu một transaction được thực hiện thành công, tất cả các thay đổi dữ liệu được thực hiện trong transaction sẽ được lưu. Nếu một transaction bị lỗi thì sẽ được rollback và thì tất cả các sửa đổi dữ liệu sẽ bị xóa (dữ liệu được khôi phục về trạng thái trước khi thực hiện transaction).
* 4 đặc điểm tiêu chuẩn của transaction
* Bảo toàn - đảm bảo rằng tất cả các câu lệnh trong nhóm lệnh được thực thi thành công
* Nhất quán - đảm bảo rằng cơ sở dữ liệu thay đổi chính xác các trạng thái khi một transaction được thực thi thành công.
* Độc lập - cho phép các transaction hoạt động độc lập và minh bạch với nhau.
* Bền bỉ - đảm bảo rằng kết quả của một transaction được commit vẫn tồn tại.

1. Các procedure transaction nghiệp vụ thực tế

* Bài toán 1 : Store thanh toán hóa đơn mua hàng của họ, nếu số tiền thiếu thì hủy giao dịch.



A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

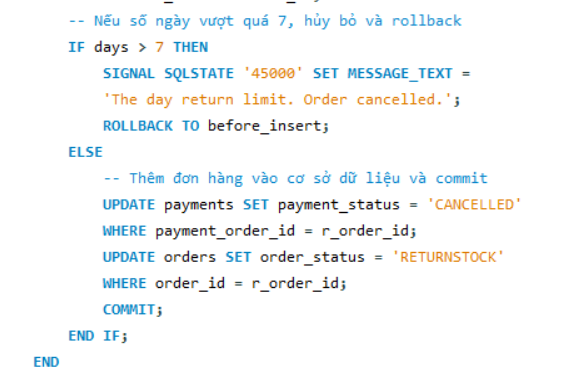
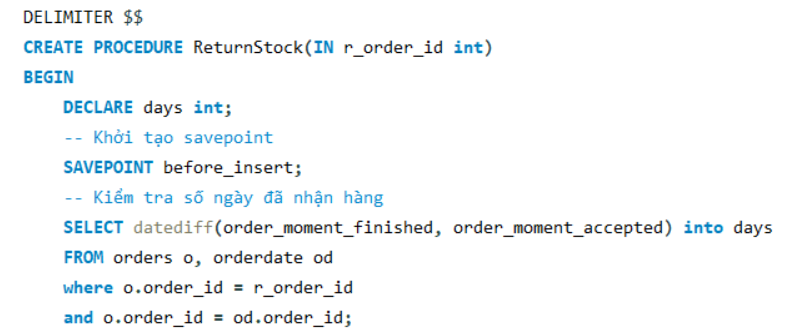
Khi thực hiện chạy chương trình :

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

* Bài toán 2 : Store đã nhận hàng nhưng vì lí do nào đó muốn trả hàng, transaction thực hiện kiểm tra xem đã quá hạn đổi trả chưa, nếu đã quá hạn thì không trả được (thường là trong 1 tuần thì được trả).

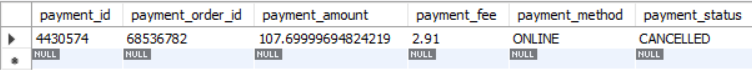


Khi thực hiện chạy chương trình :

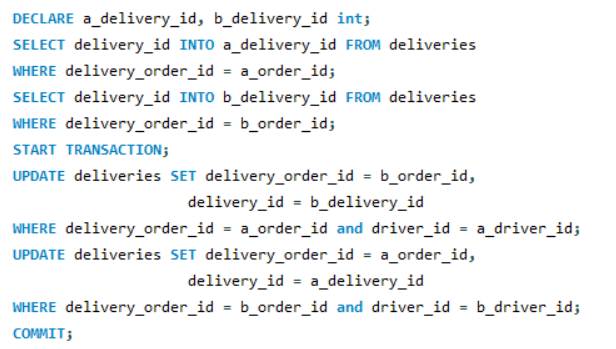
* Bảng orderdate, ta thấy ngày giao hàng và ngày nhận hàng không vượt quá 7.



* Sau khi cập nhật :



* Bài toán 3 : Anh A và B vì lí do cá nhân nên đổi đơn hàng C và D cho nhau.

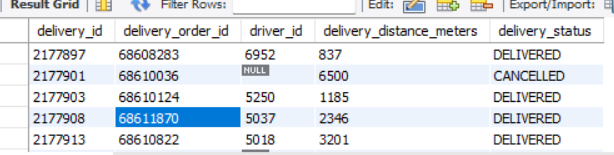
A screenshot of a computer code

Description automatically generated

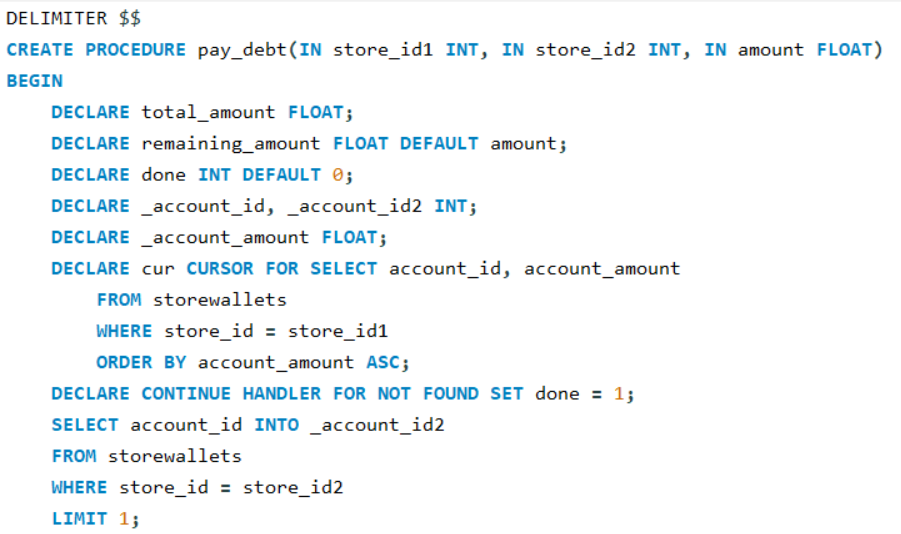
Khi thực hiện gọi chương trình :

A screenshot of a computer

Description automatically generated



* Bài toán 4: Store A nợ store B một khoản tiền, bây giờ họ muốn trả tiền nhau thông qua ví thanh toán của mình.



A computer screen shot of a code

Description automatically generated

A screenshot of a computer code

Description automatically generatedA screenshot of a computer program

Description automatically generated



Khi thực hiện chạy chương trình :

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

* Bài toán 5 : Store đặt hàng, kiểm tra xem họ có đang nợ quá số tiền được phép không, nếu không thì không cho đặt đơn.

A computer screen shot of a code

Description automatically generated

A computer screen shot of text

Description automatically generated



Khi thực hiện chạy chương trình :

* Khi họ đang nợ quá số tiền :





1. PROCEDURE ĐỔ DỮ LIỆU
2. Đổ toàn bộ dữ liệu sang CSDL mới

A screenshot of a computer code

Description automatically generatedA computer screen shot of a code

Description automatically generated

A computer screen shot of a code

Description automatically generated

Tư tưởng của chương trình :

* Sử dụng các câu lệnh SELECT dữ liệu các bảng, các tham chiếu từ system
* Sử dụng con trỏ, tạo vòng lặp để tạo ra các bảng và đổ dữ liệu
* Sau khi có dữ liệu, tạo các khóa ngoại tham chiếu
* Sử dụng các PREPARE STAMENT để tạo các câu lệnh có sử dụng tham số và biến

Khi sử dụng ta gọi chương trình, ta truyền vào tên CSDL ta muốn tạo :

Chương trình tạo ra một CSDL mới y hệt CSDL cũ, bao gồm cả các khóa ngoại :

A diagram of a computer

Description automatically generated with medium confidence

1. Đổ dữ liệu từ ngày nào đó

Tư tưởng :

* Giống như chương trình ở trên, nhưng lúc này, ta phải xem xét các trường liên quan đến ngày tháng
* Do có liên quan đến các trường ngày tháng, không thể sử dụng con trỏ và vòng lặp để câu lệnh ngắn gọn hơn vì ta không biết được các trường nào có ngày tháng khi dùng con trỏ

A computer screen shot of a code

Description automatically generatedA screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generated

A computer screen shot of a code

Description automatically generated

Khi thực hiện gọi chương trình :

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

ON DAY

ALL

1. TỰ ĐỘNG SINH PROCEDURE INSERT, UPDATE, DELETE

Sử dụng ngôn ngữ lập trình PYTHON nhúng vào MySQL, ta có thể tạo được các procedure insert, update, delete dữ liệu từng bảng khi mà CSDL quá nhiều bảng, hoặc đơn giản là không muốn tạo ra nó một cách thủ công và tốn nhiều thời gian.

* Sử dụng

A screenshot of a computer code

Description automatically generatedA screenshot of a computer code

Description automatically generated

A screenshot of a computer code

Description automatically generatedA screenshot of a computer code

Description automatically generated

A screen shot of a computer code

Description automatically generatedA screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generatedSau khi biên dịch chương trình trên, trong MySql, ta thu được procedure tương ứng với biến truyền vào là tên bảng và công việc thực thi (INSERT, UPDATE, DELETE)

MỞ RỘNG