* **Vấn đề thiết kế tập trung**

Đang phân vân:

1. **Vấn đề bảng có hay không bảng KHÁCH HÀNG**

* Hôm trước Vũ thiết kế thì có bảng KHÁCHHÀNG. Bảng này lưu thông tin chi tiết về khách hàng( Mã, họ tên, Số CMND, SĐT, Email, Địa chỉ,…)
* Nhưng bản update sau này lại xóa bảng đó, thông tin Khách hàng( Tên khách hàng, CMND, SĐT ) được thêm vào trong PHIẾU ĐẶT CHỖ.
* Thiết nghĩ, làm như vậy thì giảm được không gian lưu trữ thông tin( không đáng kể) nhưng bù lại nhà hàng sẽ khó khăn trong việc phân tích và đề xuất các kế hoạch kinh doanh. Cụ thể:

+ Trong kinh doanh thường quan tâm đến ý kiến khách hàng về chất lượng phục vụ, chất lượng món ăn,…Giả sử nhà hàng cần khảo sát lấy ý kiến để nâng cao chất lượng nhà hàng thì sẽ thiếu thông tin về khách hàng để triển khai. Nếu để những thông tin liên lạc trong bảng PHIẾU ĐẶT CHỖ thì xem ra không hợp lý.

+ Trong kinh doanh, cũng thường có các dịp khuyến mãi nhân ngày lễ. Nhà hàng cần nắm rõ sự tương quan giữa món ăn - khách hàng( lưá tuổi). VD như ngày quốc tế thiếu nhi thì nên khuyến mãi những món ăn gì ? 🡪 Sau đó gửi thông tin khuyến mãi tới khách hàng.

+ …

Vì vậy, Vũ nghĩ nên có bảng KHÁCH HÀNG để lưu trữ thông tin khách hàng nhằm phụ vụ thống kê, quan hệ khách hàng, kế hoạch kinh doanh,…

1. **Vấn đề bảng có hay không bảng PHÂN QUYỀN ?**

Trong thiết kế CSDL, Vũ nghĩ không có bảng này. Nó thiên về hướng cài đặt và vận hành hệ thống. Vì nó liên quan đến ROLE, USER, TÍNH RIÊNG TƯ,…

1. **Vấn đề thêm vào quan hệ THÔNG TIN ĐĂNG NHẬP**

* Dùng để lưu trữ thông tin về Username và Password của người dùng trong hệ thống. Trong đó, Username chính là mã nhân viên.
* **Vấn đề thiết kế phân tán**

*Ghi chú sau khi gặp thầy ngày 10/4/2012*

*1) Làm lại báo cáo theo hướng diễn giải rõ ràng hơn*

*- Chia ra từng node*

*- Giải thích ở mỗi node, mỗi bảng sẽ thay đổi cái gì? lí do thay đổi*

*- Có thể đánh dấu sự thay đổi như thế nào trên hình. Nói chung là tìm ra 1 cách trình bày vừa rõ ràng vừa thu hút và lập luận phải hợp lý, thuyết phục*

🡺 Vấn đề này nhóm phân tán nên format lại báo cáo theo cách cô đọng, dễ hiểu nhất có thể.

*2) Chú ý về thiết kế CSDL:*

*-- Phân mảnh ngang: chú ý ở chỗ khi phân tán việc đồng bộ và đặt khóa chính khó khăn (việc liên kết mấy cái client để ra được kết quả câu truy vấn sẽ như thế nào) - tìm hiểu replication - (1 khái niệm được dùng để xử lý vấn đề replication trong oracle là datalink*

*-- Phân mảnh dọc: việc đặt khóa chính toàn cục, việc đặt khóa chính và đồng bộ hóa khi phân mảnh dọc*

🡺 Góp ý ở cuối bài này.

***Các lỗ hổng thiết kế:***

*-- Nhà cung cấp: nhà cung cấp nên đặt trên client hay tổng công ty, vì*

*+ 1 là nếu đặt ở client thì client có thể dễ dàng tự quản lý nhà cung cấp cho mình*

*+ 2 là nếu đặt ở server thì tiện cho việc 1 nhà cung cấp có thể cung cấp cho nhiều nhà hàng*

🡺 Nhà cung cấp nên đặt ở 2 nơi: Tổng công ty và các nhà hàng. <Phân mảnh ngang – Phân mảnh dọc>

-- *Quản lý phiếu nhập xuất:*

*+ không nên quản lý theo phiếu nhập xuất mà quản lý theo hợp đồng*

*+ hợp đồng sẽ bao gồm thông tin thanh toán, thời hạn hợp đồng (chú ý trường hợp khi hết hàng thì mình sẽ làm ntn nếu có hợp đồng)*

🡺 Việc quản lý nhập xuất bằng hợp đồng cũng là một cách khá hay khi ràng buộc nhà cung cấp với nhà hàng bằng một loạt các thỏa thuận. Nó giải quyết được việc khi nhà hàng đặt hàng mà nhà cung cấp không có hàng thì phải giải quyết ntn ?

Đảm bảo cho nhà hàng giữ thế chủ động trong việc nhập nguyên liệu.

Tuy nhiên, không thể bỏ được quan hệ PHIẾU NHẬP HÀNG. Vì các lý do sau:

- Hợp đồng đối với nhà cung cấp thể hiện thiết lập quan hệ làm ăn giữa nhà hàng và nhà cung cấp, thông quan một số thỏa thuận ít khi thay đổi. Nếu có thay đổi thì sẽ thống nhất lại 🡪 Bản hợp đồng mới

- Trong khi đó, việc nhập hàng của Nhà hàng có sự thay đổi thường xuyên về mặt nguyên liệu, số lượng,… Không thể một ngày ký một hợp đồng, hợp đồng kí một lần( nhiều lần nếu có thay đổi), trong khi đó phiếu nhập hàng thì hàng ngày.

Vậy nên hướng của nhóm Vũ là bổ sung quan hệ HỢP ĐỒNG, CHI TIẾT HỢP ĐỒNG, và giữ lại PHIẾU NHẬP HÀNG( bổ sung 1 số thuộc tính cho phù hợp). Việc này dẫn đến sẽ lược bỏ một số thuộc tính trong quan hệ NHÀ CUNG CẤP như thuộc tính phương thức thanh toán, hình thức thanh toán, vì những cái này đã được ghi trong hợp đồng. ( có thể với mỗi hợp đồng sẽ có phương thức và hình thức thanh toán riêng đối với từng nguyên liệu, thay vì gắn chết phương thức và hình thức thanh toán đối với nhà cung cấp, trong khi nhà cung cấp có thể cung ứng nhiều loại nguyên liệu. Đâu phải nguyên liệu nào cũng có phương thức và hình thức thanh toán như nhau.)

3 Quan hệ mới bổ sung là:

**HỢPĐỒNG**(MÃHỢPĐỒNG, BÊN\_A, BÊN\_B, PHƯƠNGTHỨCTHANHTOÁN, HÌNHTHỨCTHANHTOÁN, NHÂNVIÊNPHỤTRÁCH, NGÀYKÝHỢPĐỒNG, NGÀYHẾTHẠN)

**CHITIẾTHỢPĐỒNG**(MÃHỢPĐỒNG, MÃNL, SỐLƯỢNGMAX, SỐLƯỢNGMIN, ĐƠNGIÁ, XỬLÝ)

**PHIẾUNHẬPHÀNG**(MÃHỢPĐỒNG, MÃPHIẾUNHẬP, MÃNL, SỐ LƯỢNG, GIÁTRỊPHIẾUNHẬP)

*+ hợp đồng mở rộng có thể qui định giá cả, sau 1 thời gian thì có thể thay đổi giá cả*

🡺 Cái này thể hiện trong **CHITIẾTHỢPĐỒNG**

*-- Quản lý nguyên liệu:*

*+ bỏ cái loại nguyên liệu ra khỏi csdl (vì t thấy không cần nó làm gì nữa hết) - dòng này là ý kiến cá nhân*

*+ giữ lại loại nguyên liệu thì nó sẽ chứa đại loại như thịt gà,.. thì sẽ ntn*

🡺 Ta có thể ví dụ như sau:

Thịt gà là tên nguyên liệu, loại nguyên liệu là thực phẩm tươi sống

Măng khô là tên nguyên liệu, loại nguyên liệu là thực phẩm khô

Đường, muối, ớt,..là tên nguyên liệu, loại nguyên liệu là gia vị

...

*+ nguyên liệu chứa trong kho ngoài sức chứa (là giá trị max) còn cần có giá trị min để chương trình có thể căn cứ vào đó mà báo là hết hàng hay chưa*

🡺Bổ sung:

**KHOHÀNG**(MÃNH, MÃNL, SỐLƯỢNG,TÌNHTRẠNG, SỐLƯỢNGMAX, SỐLƯỢNGMIN )

*-- Quản lý món ăn:*

*+ đặt MONAN ở server hay client trong phân tán*

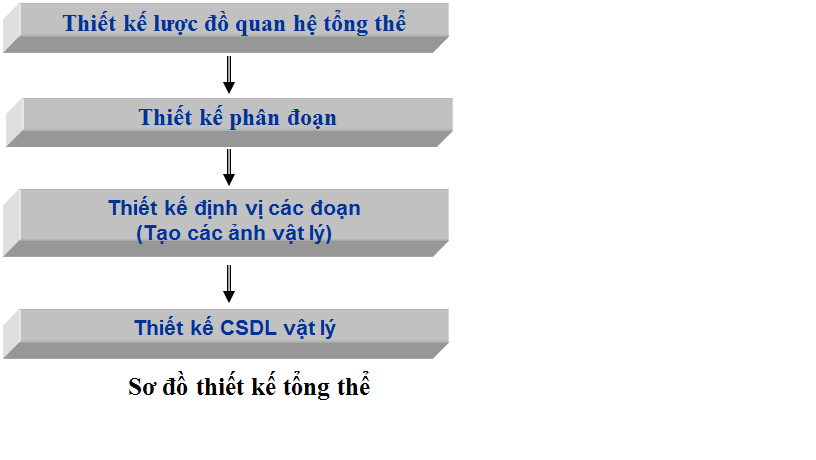
*+ chưa tối ưu ở chỗ MONAN giống nhau của nhiều nhà hàng (khi đó thì nên đặt ở server)*

*T đang phân vân là việc insert, update, delete (thường xảy ra ở client) và việc select (thường báo cáo trên server) trên cùng một bảng thì mình nên chọn việc đặt bảng đó ở đâu.*

🡺 Nhóm cần tìm hiểu hơn về kĩ thuật phân mảnh để quyết định nó đặt ở đâu.

* Note của Hà Thảo ghi sau khi gặp thầy ngày 10/3 tập trung đa số vào vấn đề thiết kế CSDL phân tán. Thực sự nếu không được giải thích từ phía tác giả mà chỉ nhìn vào báo cáo thì rất dễ làm người khác hiểu lầm và khó trong việc có cái nhìn từ tổng quan đến chi tiết vấn đề phân tán dữ liệu.
* Một vấn đề mà nhóm phân tán( 430 -431) đặt ra là một bảng nào đó nên đặt ở server hay client. Vấn đề này, thiết nghĩ, các bạn chưa hiểu kĩ về kĩ thuật phân tán dữ liệu, tạo ra lỗ hổng thiết kế nên chưa triệt để trong việc định vị quan hệ trên mạng phân tán.
* Bên nhóm tập trung( 517 - 601) xin nêu ra một số vấn đề mà nhóm hiểu về thiết kế CSDL phân tán và đề xuất một cách trình bày rõ ràng hơn so với báo cáo của nhóm bạn.

1. Mô hình thiết kế CSLD phân tán có thể dựa vào mô hình chuẩn sau:



Trong đó:

***Thiết kế lược đồ quan hệ tổng thể*:**

* + Thiết kế các quan hệ tổng thể
  + Mô tả toàn bộ dữ liệu sẽ được dùng trong hệ thống
    - Cái này chính là Lược đồ CSDL Quan hệ Tập trung

1. ***Thiết kế phân đoạn:*** thực hiện chia nhỏ dữ liệu thành các phần. Việc chia nhỏ theo 3 kĩ thuật: phân mảnh ngang , phân mảnh dọc, và phân mảnh hỗn hợp.
2. ***Thiết kế định vị các đoạn:*** Sau khi các bạn đã phân mảnh được các quan hệ, việc tiếp theo là định vị các quan hệ con vào các vị trí trạm trên hệ thống mạng phân tán.
3. ***Thiết kế cơ sở dữ liệu vật lý****:* thiết kế dữ liệu vật lý cho các quan hệ tại các trạm

Thiết kế phân tán có thể thiết kế từ trên xuống hoặc từ dưới lên. Ở đây ta nên thiết kế theo hướng từ trên xuống( tức là từ tổng thể đến riêng biệt).

**Phương pháp thiết kế từ trên xuống**

* Thiết kế từ tổng thể đến riêng biệt
* Phân rã một hệ thống lớn thành các hệ thống con
* Phân tích các yêu cầu nhằm định nghĩa môi trường hệ thống
* Thu thập các yêu cầu về dữ liệu và nhu cầu xử lý của các trạm có sử dụng CSDL.

Việc phân rã hệ thống thực chất là phân tán quan hệ và định vị chúng trên môi trường mạng. Vậy nên nó cần một số tiêu chí để làm đầu vào cho thuật toán phân tán. Cách để có yêu cầu là xét các yêu cầu dữ liệu và nhu cầu xử lý trên các thuộc tính của quan hệ, trên các record của quan hệ.

Cụ thể:

Bạn phải giả sự hệ thống nhà hàng có 3 nhà hàng thành viên và 1 tổng công ty. Việc đầu tiên là phải mô hình hóa chúng thành sơ đồ mạng. Sau đó mới phân rã từ CSDL tập trung.

Phải tiến hành trên tất cả quan hệ của CSDL tập trung.

Ví dụ như xét trên quan hệ NHANVIEN.

Trong quan hệ này, các nhu cầu xử lý thường gặp là gì :

Truy xuất thông tin nào thường diễn ra trên quan hệ nhân viên ?

Thêm nhân viên ?

Xóa nhân viên ?

Update thông tin nhân viên ?....

Sau đó dựa vào các xử lý dữ liệu đó, xem ai sẽ làm việc đó ?( đây là cơ sở để định vị dữ liệu trên các vị trí trong môi trường phân tán).

* Nếu là các nhà hàng có nhu cầu xử lý dữ liệu thì họ thường xử lý những dữ liệu gì?
* Tổng công ty có liên quan gì đến quan hệ NHANVIEN không ?
* Họ có thao tác gì trên dữ liệu của quan hệ nhân viên không ?
* Nếu không thì ta chỉ cần phân mảnh ngang, còn nếu có thì phải xem xét phân mảnh dọc nữa hay phân mảnh hỗn hợp (Vì bản chất: trên cùng quan hệ NHANVIEN, dữ liệu mà Tổng công ty thao tác khách với dữ liệu mà các NHAHANG thao tác, dựa vào cái này mà chẻ bảng thành các quan hệ con,.)
* ….

Bạn cũng cần phải biễu diễn sự thay đổi của một quan hệ từ tập trung sang phân tán. Tức là mình đã phân tán cái gì ? Trên tiêu chí nào ? Mà cách ngắn gọn và dễ hiểu nhất chính là dùng sơ đồ chuyển dịch, biễu diễn sự thay đổi của quan hệ.

Sau giai đoạn phân tán hệ thống, bạn đi đến giai đoạn định vị quan hệ. Tiếp tục với giả sử Hệ thống của bạn có 3 nhà hàng con và 1 tổng công ty, biễu diễn sự định vị các quan hệ trên sơ đồ, hoặc dùng cách nào đó, để thầy thấy được sự liên quan của các quan hệ trên hệ thống phân tán.

Một số vấn đề cần nghĩ tới, là sự truy xuất và đồng bộ hóa dữ liệu khi phân mảnh thành các quan hệ con Q1, Q2,…Qn. Làm sao để đồng bộ khi các Qi nằm trên các vị trí khách nhau . Cái này thiên về hướng lập trình xử lý, nhưng nếu không nghĩ đến cái này thì mai mốt rất khó lập trình.

V gửi các bạn thêm một số tài liệu về thiết kế CSDL phân tán, để các bạn đọc qua xem tiến trình của một Hệ thống phân tán nên bắt đầu từ đâu ? Và cần có những gì sau khi phân tán nó ? Cũng cần lưu ý một số thuật toán phân tán dữ liệu như BEA, thuật toán phân hoạch và thuật toán PARTITION