TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

──────── \* ───────

ĐỒ ÁN

TỐT NGHIỆP ĐẠI HỌC

NGÀNH CÔNG NGHỆ THÔNG TIN

Xây dựng kho dữ liệu phân phối, bán hàng trên mã nguồn mở Ofbiz

Sinh viên thực hiện: **Nguyễn Việt Anh**

Lớp CNTT1 – K55

Giáo viên hướng dẫn: TS. **Phạm Văn Hải**

HÀ NỘI 05-2015

# **PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP**

1. Thông tin về sinh viên

Họ và tên sinh viên: Nguyễn Việt Anh

Điện thoại liên lạc: 01656589111 Email: vietanh.anlao@gmail.com

Lớp: CNTT1-K55 Hệ đào tạo: Chính quy

Đồ án tốt nghiệp được thực hiện tại: Đại Học Bách Khoa Hà Nội

Thời gian làm ĐATN: Từ ngày 01/01/2015 đến 29/05/2015.

2. Mục đích nội dung của ĐATN

Xây dựng kho dữ liệu phân phối, bán hàng trên mã nguồn mở Ofbiz [Phụ lục 1].

3. Các nhiệm vụ cụ thể của ĐATN

* Thực hiện tìm hiểu Framework Ofbiz, Data model của Ofbiz
* Nghiên cứu nghiệp vụ bán hàng trong ofbiz
* Nghiên cứu về Data warehouse
* Dựa vào dữ liệu hoạt động của Ofbiz để xây dựng Data Warehouse
* Từ Data Warehouse sẽ xử lí dữ liệu để đưa ra các báo cáo

4. Lời cam đoan của sinh viên:

Tôi – *Nguyễn Việt Anh* – cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *TS. Phạm Văn Hải*.

Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

|  |  |
| --- | --- |
|  | Hà Nội, ngày 29 tháng 05 năm 2015  Tác giả ĐATN  *Nguyễn Việt Anh* |

5. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ:

|  |  |
| --- | --- |
|  | Hà Nội, ngày tháng năm  Giáo viên hướng dẫn |

# **TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP**

# **LỜI CẢM ƠN**

**Mục lục**

[**PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP** 2](#_Toc419302847)

[**TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP** 3](#_Toc419302848)

[**LỜI CẢM ƠN** 4](#_Toc419302849)

[**DANH SÁCH CÁC HÌNH** 7](#_Toc419302850)

[**BẢNG VIẾT TẮT** 8](#_Toc419302851)

[**Chương 1: Giới thiệu** 9](#_Toc419302852)

[**1.1** **Tổng quan** 9](#_Toc419302853)

[**1.2** **Phát biểu bài toán** 9](#_Toc419302854)

[**1.3** **Vấn đề đặt ra cho bài toán** 9](#_Toc419302855)

[**1.4** **Nhiệm vụ đồ án** 9](#_Toc419302856)

[**Chương 2: Cơ sở lý thuyết** 10](#_Toc419302857)

[**2.1** **Giới thiệu tổng quan về Data Warehouse** 10](#_Toc419302858)

[2.2 **Data Warehouse là gì?** 11](#_Toc419302859)

[2.3 **Đặc điểm của Data Warehouse** 12](#_Toc419302860)

[**2.4** **Các thành phần chính của Data Warehouse** 14](#_Toc419302861)

[**2.4.1** **Tầng ETL** 15](#_Toc419302862)

[2.4.2 **Tầng data warehouse** 15](#_Toc419302863)

[**2.4.3** **Tầng khai thác dữ liệu** 16](#_Toc419302864)

[2.5 **Mô hình dữ liệu thực thể của Data Warehouse** 16](#_Toc419302865)

[2.5.1  **Mô hình dạng sao** 17](#_Toc419302866)

[2.5.2 **Mô hình dữ liệu dạng bông tuyết** 18](#_Toc419302867)

[**2.6** **Kĩ thuật phân tích OLAP** 20](#_Toc419302868)

[**2.6.1** **Giới thiệu về OLAP** 20](#_Toc419302869)

[**2.6.2** **Mô hình dữ liệu đa chiều** 20](#_Toc419302870)

[**2.6.3** **Kiến trúc khối (Cube) của OLAP** 22](#_Toc419302871)

[**2.6.4** **So sánh OLAP và OLTP** 23](#_Toc419302872)

[**2.6.5** **Các thành phần của OLAP** 24](#_Toc419302873)

[2.6.6 **Các mô hình lưu trữ hỗ trợ OLAP** 24](#_Toc419302874)

[2.6.7 **So sánh các mô hình OLAP** 27](#_Toc419302875)

[**2.7** **Giới thiệu bộ công cụ Pentaho** 27](#_Toc419302876)

[**2.8** **Giời thiệu về Ofbiz** 27](#_Toc419302877)

[**2.8.1** **Giới thiệu Framework Ofbiz** 27](#_Toc419302878)

[**2.8.2** **Kiến trúc 3 tầng và cách thức phát triển ứng dụng với Ofbiz** 31](#_Toc419302879)

[**2.8.3** **Tìm hiểu thành phần và chạy ứng dụng sẵn có trên OFBIZ** 35](#_Toc419302880)

[**2.8.4** **Cấu hình để chạy Ofbiz trên localhost** 38](#_Toc419302881)

[**Chương 3: Mô tả bài toán** 40](#_Toc419302882)

[**3.1** **Tổ chức nội bộ trong công ty Delys** 40](#_Toc419302883)

[**3.2** **Quy trình xử lý đơn hàng** 42](#_Toc419302884)

[3.3 **Mô hình xây dựng Data Warehouse cho nghiệp vụ bán hàng** 44](#_Toc419302885)

[3.3.1 **Tổng quát** 44](#_Toc419302886)

[3.3.2 **Mô tả chi tiết** 45](#_Toc419302887)

[**Chương 4: Phân tích thiết kế hệ thống** 48](#_Toc419302888)

[**Chương 5: Cài đặt và kiểm thử hệ thống** 49](#_Toc419302889)

**DANH SÁCH CÁC HÌNH**

[Hình 2. 1 Luồng dữ liệu trong data warehouse 12](#_Toc419302811)

[Hình 2. 2 Các thành phần trong data warehouse 14](#_Toc419302812)

[Hình 2. 3 Mô hình dạng sao 17](file:///D:\Workspace\project\project\doc\Data%20warehouse.docx#_Toc419302813)

[Hình 2. 4 Mô hình dữ liệu dạng bông tuyết 19](file:///D:\Workspace\project\project\doc\Data%20warehouse.docx#_Toc419302814)

[Hình 2. 5 Mô hình dữ liệu đa chiều 21](#_Toc419302815)

[Hình 2. 6 Mô hình MOLAP 25](#_Toc419302816)

[Hình 2. 7 Mô hình ROLAP 26](#_Toc419302817)

[Hình 2. 8 Mô hình HOLAP 27](#_Toc419302818)

[Hình 2. 9 Các thành phần ERP 28](#_Toc419302819)

[Hình 2. 10 Kiến trúc Ofbiz 29](#_Toc419302820)

[Hình 2. 11 Phát triển ứng dụng Java 32](#_Toc419302821)

[Hình 2. 12 Phát triển ứng dụng php 33](#_Toc419302822)

[Hình 2. 13 Phát triển ứng dụng Ofbiz 34](#_Toc419302823)

[Hình 2. 14 Tổ chức thư mục Ofbiz 37](#_Toc419302824)

[Hình 2. 15 Cấu hình CDSL trong Ofbiz 39](#_Toc419302825)

[Hình 2. 16 Lưu trữ driver jdbc trong Ofbiz 39](#_Toc419302826)

[Hình 2. 17 Tạo component trong Ofbiz 40](#_Toc419302827)

[Hình 3. 1 Tổ chức nội bộ Delys 41](#_Toc419302828)

[Hình 3. 2 Cấu trúc bảng Party Relationship trong Ofbiz 42](#_Toc419302829)

[Hình 3. 3 Dữ liệu thực tế Delys trong Party Relationship 42](#_Toc419302830)

[Hình 3. 4 Quy trình xử lý đơn hàng tại Delys 43](#_Toc419302831)

[Hình 3. 5 Quan hệ các bảng trong Ofbiz 44](#_Toc419302832)

[Hình 3. 6 Mô hình data warehouse 45](#_Toc419302833)

[Hình 3. 7 Quá trình xử lý dữ liệu trong các vùng DW 46](#_Toc419302834)

[Hình 3. 8 Cập nhật dữ liệu vào DW 47](#_Toc419302835)

[Hình 4. 1 Biểu đồ Use case tổng quan 49](#_Toc419302836)

# **BẢNG VIẾT TẮT**

|  |  |
| --- | --- |
| OLAP | Online Analysis Processing |
| MOLAP | Multidimensional Online Analysis Processing |
| HOLAP | Hybird Online Analysis Processing |
| BI | Business Intelligence |
| OLAP | OnLine Transaction Processing |
| ROLAP | Relational Online Analysis Processing |

# **Chương 1: Giới thiệu**

* 1. **Tổng quan**

Mọi doanh nghiệp để có được sự thành công của mình, nhất thiết phải cần vào sự quyết định đúng đắn của các nhà quản trị của doanh nghiệp đó - việc quyết định của nhà quản trị phải dựa trên các số liệu liên quan, và một minh chứng khoa học để đi đến quyết định đó, và cách tốt nhất để có nhanh được các dữ liệu liên quan, và tìm kiếm các minh chứng khoa học này là nhà quản trị phải có một kho dữ liệu được sắp xếp một cách khoa học. Vì vậy trong đồ án này em xin đề xuất việc xây dựng Data Warehouse-kho dữ liệu cho các doanh nghiệp, nhà quản trị… trong việc kinh doanh sản phẩm.

* 1. **Phát biểu bài toán**

Bài toán được phát biểu như sau: Doanh nghiệp hiện tại đang thực hiện triển khai ERP trên công nghệ mã nguồn mở OFBIZ, cần xây dựng một kho dữ liệu phục vụ báo cáo, thống kê liên quan đến nghiệp vụ phân phối,bán hàng.

* 1. **Vấn đề đặt ra cho bài toán**

Những vấn đề đặt ra cho bài toán được phân tích như sau**:**

* Hiện tại doanh nghiệp đang triển khai quản lý trên FrameWork Ofbiz vì vậy cần thực hiện tìm hiểu luồng dữ liệu bán hàng để xây dựng cấu trúc của Data Warehouse.
* Xây dựng cấu trúc cho kho dữ liệu từ OFbiz
* Quản lý, lập lịch thực hiện việc cập nhật dữ liệu cho Data Warehouse từ dữ liệu hoạt động của doanh nghiệp, đảm bảo không ảnh hưởng đến quá trình hoạt động của doanh nghiệp.
* Tạo các báo cáo, biểu đồ trên Ofbiz lấy dữ liệu từ kho dữ liệu phù hợp với từng vị trí, cấp bậc người sử dụng
* Xây dựng giao diện thân thiện và dễ sử dụng với người dùng.
  1. **Nhiệm vụ đồ án**

Nhiệm vụ của đồ án: Xây dựng kho dữ liệu dựa trên mã nguồn mở Ofbiz phục vụ nhu cầu báo cáo, thống kê cho doanh nghiệp.

# **Chương 2: Cơ sở lý thuyết**

* 1. **Giới thiệu tổng quan về Data Warehouse**

Trên thế giới có rất nhiều tổ chức hay công ty có những kho dữ liệu khổng lồ. Trong quá trình hoạt động, do yêu cầu của công việc nên những tổ chức hay công ty này phải lưu lại dữ liệu của khách hàng, sản phẩm, hóa đơn, ... Theo nghiên cứu thì cứ khoảng sau 5 năm khối lượng dữ liệu của doanh nghiệp lại tăng lên gấp đôi. Tuy nhiên việc làm thế nào để có thể khai thác được những kho dữ liệu khổng lồ này lại là một vấn đề nan giải bởi dữ liệu thường không nhất quán và không được thiết kế với mục đích hỗ trợ quyết định. Vì vậy cái mà các tổ chức và công ty này cần là một công cụ cho phép họ tổng hợp dữ liệu một cách hiệu quả.

Sự ra đời của data warehouse bắt nguồn từ sự kết hợp của hai nhân tố kinh tế và kĩ thuật. Các công ty, đặc biệt là các công ty lớn có nhiều chi nhánh khác nhau, rất muốn có được những thông tin tổng quan trên phạm vi toàn công ty, từ đó có thể giúp cho lãnh đạo của công ty ra quyết định. Tuy nhiên, các hệ thống dữ liệu tác nghiệp trước đây không thể hoàn thành được công việc này vì chúng được thiết kế với mục đích là phục vụ công việc nghiệp vụ hàng ngày, vì vậy cần phải có một công nghệ dữ liệu khác. Chính sự phát triển của công nghệ được sự thúc đẩy mạnh mẽ của nhu cầu của các doanh nghiệp trong môi trường kinh doanh đầy cạnh tranh đã tạo nên data warehouse.

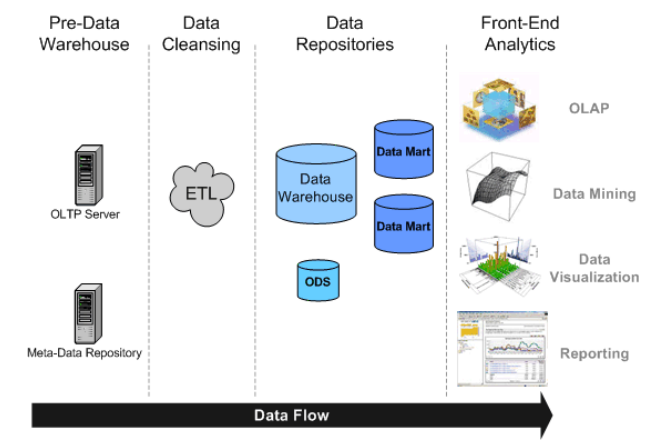
Data warehouse ra đời đã mang lại một số lợi ích nhất định cho các nhà kinh doanh :

* Cho phép các nhà kinh doanh theo dõi được sự biến động của môi trường kinh doanh : ở một khía cạnh nào đó, data warehouse có thể xem như là những "ảnh chụp" liên tiếp tình trạng kinh doanh của công ty, vì vậy nó cho phép người dùng cuối thấy được môi trường kinh doanh của công ty qua các thời kì.
* Câu trả lời cho các câu hỏi truy vấn dữ liệu có tính thương mại. Các nhà kinh doanh muốn rằng những câu hỏi họ đặt ra trong lĩnh vực kinh doanh sẽ được trả lời theo dạng mà họ có thể hiểu được.
* Cung cấp công cụ truy xuất dữ liệu nhằm mục đích phân tích và hỗ trợ quyết định. Trong những hệ thống cơ sở dữ liệu tác nghiệp trước đây thì điều này rất khó thực hiện ngay cả khi dữ liệu là đầy đủ và sẵn sàng, lÝ do đơn giản là vì hệ thống cơ sở dữ liệu tác nghiệp được thiết kế ra với mục đích phục vụ công việc tác nghiệp thường ngày chứ không phải được thiết kế ra với mục đích hỗ trợ quyết định.
* Thống nhất được dạng và cấu trúc dữ liệu phù hợp với yêu cầu của người dùng cuối. Đối với các công ty có nhiều chi nhánh khác nhau thì có thể dữ liệu được lưu trữ dưới các dạng khác nhau, ví dụ như cùng là đơn vị đo chiều dài nhưng có nơi dùng đơn vị là cm, có nơi dùng đơn vị là inch, ... do vậy gây khó khăn cho việc phân tích dữ liệu để hỗ trợ quyết định.
* Quản trị dễ dàng. Sau khi được xây dựng, người quản trị có thể lên kế hoạch để hệ thống vận hành tự động. Người quản trị chỉ có nhiệm vụ theo dõi, chỉnh sửa và nâng cấp khi có yêu cầu hoặc sự cố.
  1. **Data Warehouse là gì?**

Là một CSDL hướng đối tượng được thiết kế với việc tiếp cận các ý kiến trong mọi lĩnh vực kinh doanh. Nó cung cấp các công cụ để đáp ứng thông tin cần thiết cho các nhà quản trị kinh doanh tại mọi cấp độ tổ chức với những yêu cầu dữ liệu phức hợp và lấy thông tin nhanh, chính xác.

Data warehouse được sử dụng như là một nền tảng cho hệ thống hỗ trợ ra quyết định. Nó gắn liền với khái niệm Business Inteligence (Kinh doanh thông minh). Nó được dùng để giải quyết các vấn đề gặp phải khi một tổ chức cố gắng phân tích chiến lược từ số liệu trong một hệ thống database được dùng chung với hệ thống xử lý dữ liệu trực tuyến (Online Transaction Processing - OLTP).

Một hệ thống OLTP điển hình được đặc trưng bởi một hệ thống có nhiều người sử dụng đồng thời để thêm, sửa, xóa dữ liệu. Ví dụ như một hệ thống bán lẻ, có nhiều nhân viên bán hàng đồng thời cùng nhập và chỉnh sửa dữ liệu vào hệ thống. Trong một hệ thống nhỏ, việc sử dụng cùng một hệ thống database này cho mục đích lên báo cáo thống kê, phân tích có thể chấp nhận được do khối lượng dữ liệu ít, ít người dùng. Nhưng trong một hệ thống với khối lượng dữ liệu hàng chục triệu đến trăm triệu bản ghi trên một bảng thì việc khai thác dữ liệu chung trên cùng một hệ thống database OLTP là không thể chấp nhận được vì thời gian xử lý dữ liệu rất lâu, làm cho tài nguyên hệ thống trở nên quá tải và có thể làm cho toàn bộ hệ thống bị tê liệt.



Hình 2. 1 Luồng dữ liệu trong data warehouse

Hệ thống OLTP được thiết kế cho mục đích thu thập dữ liệu, khối lượng dữ liệu càng càng càng lớn khiến cho nhu cầu phân tích càng trở nên cấp thiết hơn. Tuy nhiên database trong OLTP được thiết kế cho mục đích thu thập số liệu thường phải tối ưu hóa cả về không gian lưu trữ lẫn chuẩn hóa trong thiết kế tuân thủ nghiêm ngặt theo mô hình dữ liệu quan hệ. Còn khi nhu cầu phân tích, khai thác dữ liệu thông qua các báo cáo sẽ phát sinh hàng loạt các vấn đề:

- Người xây dựng báo cáo không hiểu mối quan hệ phức tạp giữa các bảng trong một loạt database của các ứng dụng khác nhau.

- Database của các ứng dụng có thể để rải rác trên nhiều server, rất khó cho việc tìm kiếm các bảng để xây dựng câu truy vấn.

- Việc phân quyền không cho phép người dùng có thể lấy dữ liệu chi tiết.

- DBA hạn chế việc chạy các câu query lớn nhằm ngăn chặn nguy cơ làm cho hệ thống bị tê liệt.

Bằng cách copy dữ liệu từ một hệ thống OLTP sang một máy chủ theo một lịch trình cho mục đích báo cáo, tổ chức có thể cải thiện thời gian phản hồi cho báo cáo và các câu truy vấn. Một database thiết kế trong hệ thống OLTP là không đủ linh hoạt cho một ứng dụng hỗ trợ ra quyết định, lý do chủ yếu do khối lượng dữ liệu quá lớn và sự phức tạp của các bảng trong các database có mô hình quan hệ.

* 1. **Đặc điểm của Data Warehouse**

***a) Hướng chủ đề :***

Data warehouse được thiết kế để giúp bạn phân tích dữ liệu nhằm trả lời các câu hỏi của người dùng cuối. Mặt khác, những câu hỏi của người dùng cuối lại có thể phân loại được theo một số chủ đề nào đó. Ví dụ người dùng cuối có thể đặt các câu hỏi như : "Khách hàng có độ tuổi nào mua nhiều bảo hiểm nhất trong năm nay ?", "Công ty con nào bán được nhiều bảo hiểm nhất trong tháng này ?" hay "Mặt hàng bảo hiểm nào được ưa chuộng nhất trong quý vừa rồi ?" để tìm hiểu rõ tình trạng bán bảo hiểm của công ty. Để trả lời những câu hỏi này, một data warehouse tập trung vào chủ đề hợp đồng bảo hiểm sẽ được xây dựng. Tương tự như vậy, data warehouse hướng vào các chủ đề khác cũng sẽ được hình thành.

***b) Tích hợp :***

Data warehouse được xây dựng từ nhiều nguồn dữ liệu khác nhau, các nguồn dữ liệu này có sự xung đột với nhau về tên trường, giá trị thể hiện, đơn vị đo lường, ... Ví dụ như cùng biểu diễn trường địa chỉ nhưng có cơ sở dữ liệu sử dụng tên trường là "Địa chỉ" trong khi cơ sở dữ liệu khác lại dùng là "Nơi ở". Ngoài ra các cơ sở dữ liệu cũng có thể có giá trị thể hiện khác nhau, ví dụ để biểu diễn giới tính của khách hàng, công ty con này dùng giá trị là 0 và 1, trong khi công ty con khác lại dùng giá trị là "M" và "W". Ngoài ra cũng còn phải kể đến sự sử dụng đơn vị đo lường khác nhau cũng dẫn tới việc khó thống nhất các cơ sở dữ liệu lại với nhau. Tóm lại, khi giải quyết được tất cả những sự xung đột dữ liệu này để dữ liệu có thể chuyển vào data warehouse theo một định dạng duy nhất, ta nói rằng chúng đã được tích hợp.

***c) Không thay đổi :***

Đặc tính này có ý nghĩa là khi dữ liệu đã được cập nhật vào data warehouse rồi thì dữ liệu không nên thay đổi nữa. Điều này cũng logic vì mục đích của data warehouse là cho phép bạn phân tích những gì đã xảy ra.

***d) Thời biến :***

Để phát hiện ra xu hướng kinh doanh, các nhà phân tích cần một khối lượng dữ liệu cực kì lớn, điều này trái ngược hẳn với hệ thống xử lý giao dịch trực tuyến, nơi mà tốc độ xử lý được đặt lên quan trọng hàng đầu còn thông tin lịch sử thì không được coi trọng. Thời biến ở đây có ý nghĩa là data warehouse phản ánh được sự thay đổi kinh doanh theo thời gian.

* 1. **Các thành phần chính của Data Warehouse**

Một hệ thống data warehouse bao gồm 3 thành phần chính sau:

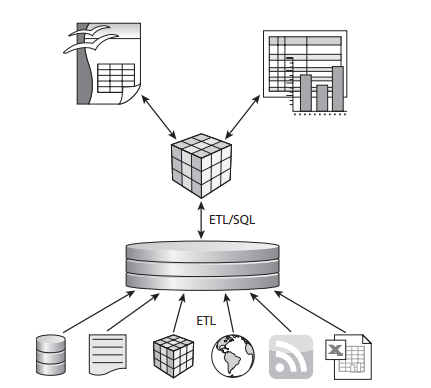
- Một bộ công cụ để thu thập dữ liệu từ hệ thống nghiệp vụ, chuẩn hoá chúng về

định dạng dữ liệu đa chiều, nạp vào data warehouse (Extract-TransformationLoading – ETL).

- Một database dùng làm data warehouse để lưu trữ dữ liệu

- Một loạt các công cụ khai thác dữ liệu từ data warehouse như hệ thống OLAP, hệ

thống báo cáo tĩnh, hệ thống data mining…



Hình 2. 2 Các thành phần trong data warehouse

* + 1. **Tầng ETL**

Tầng ETL (Extract – Transform – Load) là tầng thấp nhất, ẩn đi với người dùng

cuối, bao gồm 3 bước:

- Bước thu thập (extract) gom góp dữ li ệu từ nhiều khác nhau về. Các nguồn này

có thể là database hệ thống nghiệp vụ (MS SQL, mySQL, Oracle, DB2…), cũng

có thể là file ở các định dạng khác nhau (CSV, fix-length, excel, XML…), có thể

là dữ liệu nội bộ doanh nghiệp hoặc từ bên ngoài. Một hệ thống ETL tốt phải đả m

bảo tương thích với các nguồn dữ liệu thông dụng này.

- Bước chuẩn hoá (transform) biến đổi dữ liệu từ định dạng nguồn sang định dạng

của data warehouse (định dạng dữ liệu đa chiều đã nói ở bước trên), bao gồm các

bước nhỏ:

- Bước dọn dẹp (cleaning) xoá các bản ghi bị sai, lỗi và chuyển hoá dữ li ệu

về định dạng chuẩn chung.

- Bước tập hợp (integration) cắt gọt dữ li ệu có chung ý nghĩa từ nhiều

nguồn khác nhau về một khung duy nhất.

- Bước tổng hợp (aggregation) tổng hợp dữ li ệu dựa vào độ chi tiết c ủa data

warehouse.

- Bước nạp dữ liệu (load) ghi dữ li ệu đã được chuẩn hoá vào data warehouse. Bước

này bao gồm cả quá trình cập nhật thay đổi từ hệ thống nghiệp vụ vào data

warehouse, đảm bảo số li ệu báo cáo luôn được cập nhật. Tuỳ thuộc vào chính sách

* + 1. **Tầng data warehouse**

Tầng data warehouse đứng ở trung tâm một hệ thống data warehouse làm nhiệm

vụ lưu trữ dữ liệu bao quanh tất cả các hoạt động nghiệp vụ, các phòng ban của doanh

nghiệp. Data warehouse thường bao gồm một hoặc nhiều data mart, với data mart chính

là data warehouse thu nhỏ tập trung vào một nghiệp vụ nhất định nào đó của doanh nghiệp (ví dụ data mart về sale, data mart về kho bãi, data mart về nhân sự…)

Ngoài nhiệm vụ lưu trữ dữ liệu, tầng data warehouse còn có một thành phần khác

rất quan trọng gọi là siêu dữ liệu (metadata). Siêu dữ liệu lại được chia làm 2 nhóm là

nhóm siêu dữ liệu kỹ thu ật và siêu dữ liệu nghiệp vụ. Siêu dữ liệu nghiệp vụ(business metadata) mô tả ý nghĩa dữ liệu, các luật và ràng buộc tác động lên dữ liệu. Siêu dữ liệu

kỹ thuật (technical metadata) mô tả cách thức tổ chức, lưu trữ và điều khiển dữ liệu

trong hệ thố ng máy tính. Trong phạm vi data warehouse, siêu dữ liệu kỹ thuật được sử dụng để mô tả thông tin về data warehouse, về dữ liệu nguồn và các tiến trình ETL. Cụ thể:

- Siêu dữ liệu mô tả cấu trúc data warehouse và các data mart ở mức logic (mô tả

bảng fact, bảng dimension, cây phân cấp, nguồn gốc dữ liệu) và mức vật lý (cấu

trúc bảng, index, partition). Ngoài ra nó còn chứa thông tin bảo mật dữ liệu (xác

thực, phân quyền người dùng) và các thông tin giám sát (thống kê hiệu năng sử

dụng, báo cáo lỗi…)

- Siêu dữ liệu mô tả dữ liệu nguồn, cũng ở mức logic (cách thức và tham số kết nối

lấy dữ liệu, tần suất cập nhật dữ liệu, ý nghĩa dữ liệu) và vật lý (cấu trúc dữ liệu)

- Siêu dữ liệu mô tả các tiến trình ETL, bao gồm cả gốc gác dữ liệu (truy được dữ

liệu trên data warehouse về đến gốc gác của nó trong hệ thống nghiệp vụ), các luật

thu thập, làm sạch, chuyển hoá dữ liệu.

* + 1. **Tầng khai thác dữ liệu**

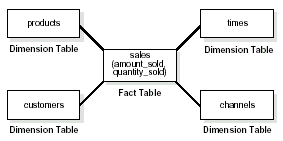
Tầng khai thác dữ liệu chứa các công cụ cho người dùng cuối khai thác, sử dụng các dữ liệu trong data warehouse. Một số công cụ chính:

* Báo cáo OLAP (OLAP tool) là báo cáo động cho phép người dùng sử dụng các tính năng c ủa OLAP (đã nói ở phần 1.3.3) để tạo báo cáo. Các truy vấn đột xuất này được gọi là truy vấn tuỳ biến (ad hoc query) vì hệ thống không hề được chuẩn bị trước cho thao tác của người dùng. Báo cáo OLAP được sử dụng khi người dùng muốn các thông tin cắt lớp, chuyên sâu hoặc toàn cảnh trước khi ra quyết đ ịnh
* Báo cáo tĩnh (reporting tool) là các báo cáo có cấu trúc, format, sử dụng truy vấn được định nghĩa trước đó, đôi khi bao gồm cả biểu đồ. Báo cáo tĩnh được sử dụng khi người dùng muốn xem các thông tin đánh giá, điều hành.
* Bộ công cụ khai phá dữ liệu (data mining) cho phép người dùng phân tích dữ

li ệu để tìm ra các thông tin quý giá còn bị ẩn dấu, ví dụ như các xu hướng, các mẫu chung

* 1. **Mô hình dữ liệu thực thể của Data Warehouse**

Cơ sở dữ liệu quan hệ đã tồn tại và phát triển trong một thời gian dài, nó đã chứng tỏ được thế mạnh của mình và thống lĩnh được thị trường, vì vậy hầu hết các hệ quản trị cơ sở dữ liệu đều sử dụng hoặc có nhân là công nghệ dữ liệu quan hệ. Tuy nhiên, để xây dựng được data warehouse thì dữ liệu cần phải là đa chiều, vì vậy ta có một số mô hình dữ liệu sau của data warehouse :

* + 1.  **Mô hình dạng sao**

Hình 2. 3 Mô hình dạng sao

Mô hình dạng sao, như cái tên của nó, có dạng hình sao. Trung tâm là một bảng sự kiện (fact table), xung quanh là các bảng chiều (dimension table) trỏ vào nó. Thông thường mỗi vùng chủ đề trong data warehouse thường xoay quanh một bảng sự kiện trung tâm. Các bảng vệ tinh xung quanh là các nhánh của bảng sự kiện có nhiệm vụ mô tả chi tiết hơn thành phần sự kiện trong bảng. Bảng sự kiện được phát hiện bằng cách phân tích công việc kinh doanh của công ty. Với mỗi công việc hoặc một tiến trình chính của công ty thường sẽ có một bảng sự kiện mô tả chi tiết tiến trình đó.

Trong ví dụ mô hình dạng sao đơn giản ở trên thì bảng sự kiện cho biết số lượng hàng bán ra và số tiền thu được ứng với các giá trị chiều là thời gian, loại sản phẩm, khách mua hàng và kênh phân phối hàng. Với mô hình như thế này thì ta có thể tính toán rất nhanh các thông số tổng hợp về số lượng bán hàng và tiền thu được, bên cạnh đó nó cũng cho ta biết được tình hình kinh doanh của công ty trong một thời gian dài. Bảng sự kiện tham chiếu đến các bảng khác thông qua các liên kết sử dụng khóa chính và khóa ngoại mà chúng ta đã rất quen thuộc.

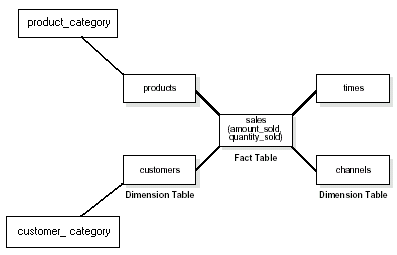
Như vậy, với mỗi thay đổi về giá trị tại các bảng chiều sẽ cho ta một giá trị mới trong bảng sự kiện. Chẳng hạn, nếu khách hàng mua hai sản phẩm trong cùng một lần thì sẽ có hai dòng trong bảng sự kiện mô tả giao dịch này. Mỗi dòng sẽ chứa lượng hàng bán và lượng tiền thu về, đồng thời tham chiếu đến các dòng chứa loại sản phẩm tương ứng trong bảng chiều sản phẩm. Nói chung là bảng sự kiện thường có dung lượng rất lớn và tăng kích cỡ liên tục, vì vậy khi thiết kế cần chú ý tới điều này.

Vậy thiết kế dữ liệu theo mô hình dạng sao rất khác với mô hình dữ liệu truyền thống vì rõ ràng mô hình dữ liệu truyền thống không cần bộ nhớ lưu trữ dữ liệu nhiều đến thế. Khác biệt này là do mục tiêu của data warehouse là cho phép người dùng cuối tìm kiếm, khai thác và xử lý thông tin một cách nhanh nhất. Người dùng cuối sẽ không quan tâm đến vấn đề giao dịch đó được diễn ra như thế nào, cái mà họ muốn biết là kết quả tổng hợp của chúng như : lượng hàng bán ra được theo vùng, doanh thu theo kì hay phân bố của nhóm khách hàng cùng mua một loại sản phẩm nào đó ... Để trả lời những câu hỏi kiểu như vậy cần một khối lượng dữ liệu rất lớn và cần thành lập những câu hỏi truy vấn phức tạp nếu ta vẫn cứ sử dụng mô hình dữ liệu kiểu truyền thống. Lúc đó vấn đề tốc độ sẽ trở nên quan trọng. Hệ quản trị cơ sở dữ liệu sẽ không thể đáp ứng được cùng một lúc nhiều câu hỏi truy vấn kiểu như vậy tại một thời điểm. Vì vậy cần tập hợp dữ liệu lại theo cấu trúc mà người dùng cần chứ không phải theo kiểu mô tả chính xác nhất các hoạt động của hệ thống.

* + 1. **Mô hình dữ liệu dạng bông tuyết**

Thực chất mô hình này chỉ là một biến thể của mô hình dạng sao. Thay vì các cánh sao chỉ gồm một bảng chiều thì các bảng chiều này lại được phân cấp thêm.

Ví dụ như các bảng chiều sản phẩm và khách hàng muốn phân cấp thêm thành loại sản phẩm và loại khách hàng.



Hình 2. 4 Mô hình dữ liệu dạng bông tuyết

Điểm khác biệt cơ bản giữa mô hình dạng bông tuyết và dạng sao là ở chỗ các chiều trong mô hình dạng bông tuyết được tách thành các cấp khác nhau. Nhờ vậy khối lượng cần lưu giữ sẽ được giảm đi. Tuy nhiên đó không phải là lợi ích chính của mô hình dạng bông tuyết. Lợi ích chính của mô hình dạng này là ở chỗ nó tạo ra một cái nhìn đầy đủ về phân cấp trong các chiều. Với mô hình này mỗi chiều được phân thành nhiều cấp và vì vậy việc khai thác dữ liệu ở mức chi tiết cũng như tổng hợp dữ liệu dễ dàng hơn. Người dùng không phải cố gắng hình dung các cấp trong mỗi chiều mà tự bản thân nó đã được mô hình hóa theo kiểu này.

Trở ngại lớn nhất khi triển khai mô hình dạng bông tuyết là vấn đề tốc độ. Đáng tiếc là nếu càng có nhiều quan hệ thì tốc độ truy vấn dữ liệu lại càng chậm, mà tốc độ lại là vấn đề sống còn khi thiết kế data warehouse. Do vậy, thông thường người ta tìm cách tránh sử dụng mô hình dạng bông tuyết. Tuy nhiên, đối với những người vốn đã quen với mô hình dữ liệu quan hệ và việc chuẩn hóa thì mô hình dạng bông tuyết gần gũi hơn và trên thực tế nhiều người thích sử dụng chúng hơn.

Cần phải nói lại rằng tiêu chí cho việc lùa chọn mô hình là tốc độ và dung lượng bộ nhớ lưu trữ. Ta có thể biến một mô hình dạng bông tuyết về dạng sao và ngược lại mà không làm ảnh hưởng gì tới ngữ nghĩa của dữ liệu. Nếu một khi đặt vấn đề tốc độ lên hàng đầu thì ta nên chọn mô hình dạng sao.

* 1. **Kĩ thuật phân tích OLAP**
     1. **Giới thiệu về OLAP**

OLAP là một kỹ thuật sử dụng các thể hiện dữ liệu đa chiều gọi là các khối (cube) nhằm cung cấp khả năng truy xuất nhanh đến dữ liệu của kho dữ liệu. Tạo khối (cube) cho dữ liệu trong các bảng chiều (dimension table) và bảng sự kiện (fact table) trong kho dữ liệu và cung cấp khả năng thực hiện các truy vấn tinh vi và phân tích cho các ứng dụng client. Trong khi kho dữ liệu lưu trữ dữ liệu cho phân tích, thì OLAP là kỹ thuật cho phép các ứng dụng client truy xuất hiệu quả dữ liệu này. OLAP cung cấp nhiều lợi ích cho người phân tích, cho ví dụ như:

- Cung cấp mô hình dữ liệu đa chiều trực quan cho phép dễ dàng lựa chọn, định hướng và khám phá dữ liệu.

- Cung cấp một ngôn ngữ truy vấn phân tích, cung cấp sức mạnh để khám phá các mối quan hệtrong dữ liệu kinh doanh phức tạp.

- Dữ liệu được tính toán trước đối với các truy vấn thường xuyên nhằm làm cho thời gian trả lời rất nhanh đối với các truy vấn đặc biệt.

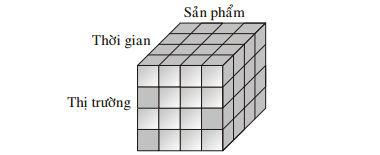
- Cung cấp các công cụmạnh giúp người dùng tạo các khung nhìn mới của dữ liệu dựa trên một tập các hàm tính toán đặc biệt.

OLAP được đặt ra để xử lý các truy vấn liên quan đến lượng dữ liệu rất lớn mà nếu cho thực thi các truy vấn này trong hệ thống OLTP sẽ không thể cho kết quả hoặc sẽ mất rất nhiều thời gian.

* + 1. **Mô hình dữ liệu đa chiều**

Các nhà quản lý kinh doanh có khuynh hướng suy nghĩ theo “nhiều chiều” (multidimensionally). Ví dụ như họ có khuynh hướng mô tả những gì mà công ty làm như sau: “Chúng tôi kinh doanh các sản phẩm trong nhiều thị trường khác nhau, và chúng tôi đánh giá hiệu quả thực hiện của chúng tôi qua thời gian”. Những người thiết kế kho dữ liệu thường lắng nghe cẩn thận những từ đó và họ thêm vào những nhấn mạnh đặc biệt của họ như: “Chúng tôi kinh doanh các sản phẩm trong nhiều thị trườngkhác nhau, và chúng tôi đánh giá hiệu quả thực hiện của chúng tôi qua thời gian”.

Suy nghĩ một cách trực giác, việc kinh doanh như một khối(cube) dữ liệu, với các nhãn trên mỗi cạnh của khối (xem hình bên dưới). Các điểm bên trong khối là các giao điểm của các cạnh. Với mô tả kinh doanh ở trên, các cạnh của khối là Sản phẩm, Thị trường, và Thời gian. Hầu hết mọi người đều có thể nhanh chóng hiểu và tưởng tượng rằng các điểm bên trong khối là các độ đo hiệu quả kinh doanh mà được kết hợp giữa các giá trị Sản phẩm, Thị trường và Thời gian:



Hình 2. 5Mô hình dữ liệu đa chiều

Mô phỏng các chiều trong kinh doanh Một khối dữ liệu (datacube) thì không nhất thiết phải có cấu trúc 3 chiều (3-D), nhưng về cơ bản là có thể có N chiều (N-D). Những cạnh của khối được gọi là các chiều (dimensions), mà đó là các mặt hoặc các thực thể ứng với những khía cạnh mà tổ chức muốn ghi nhận. Mỗi chiều có thể kết hợp với một bảng chiều (dimension table) nhằm mô tả cho chiều đó. Ví dụ, một bảng chiều của Sản phẩm có thể chứa những thuộc tính như Ma\_sanpham, Mo\_ta, Ten\_sanpham, Loai\_SP,… mà có thể được chỉ ra bởi nhà quản trị hoặc các nhà phân tích dữ liệu. Với những chiều không được phân loại, như là Thời gian, hệ thống kho dữ liệu sẽ có thể tự động phát sinh tương ứng với bảng chiều (dimension table) dựa trên loại dữ liệu. Cần nói thêm rằng, chiều Thời gian trên thực tế có ý nghĩa đặc biệt đối với việc hỗ trợ quyết định cho các khuynh hướng phân tích. Thường thì nó được mong muốn có một vài tri thức gắn liền với lịch và những mặt khác của chiều thời gian. Hơn nữa, một khối dữ liệu trong kho dữ liệu phần lớn được xây dựng để đo hiệu quả của công ty. Do đó một mô hình dữ liệu đa chiều đặc thù được tổ chức xung quanh một chủ đề mà được thể hiện bởi một bảng sự kiện (fact table) của nhiều độ đo số học (là các đối tượng của phân tích). Ví dụ, một bảng sự kiện có thể chứa số mặt hàng bán, thu nhập, tồn kho, ngân sách,… Mỗi độ đo số học phụ thuộc vào một tập các chiều cung cấp ngữ cảnh cho độ đo đó.

Vì thế, các chiều kết hợp với nhau được xem như xác định duy nhất độ đo, là một giá trị trong không gian đa chiều. Ví dụ như một kết hợp của Sản phẩm, Thời gian, Thị trường vào 1 thời điểm là một độ đo duy nhất so với các kết hợp khác. Các chiều được phân cấp theo loại. Ví dụ như chiều Thời gian có thể được mô tả bởi các thuộc tính như Năm, Quý, Tháng và Ngày. Mặt khác, các thuộc tính của một chiều có thể được tổ chức vào một lưới mà chỉ ra một phần trật tự của chiều. Vì thế, cũng với chiều Thời gian có thể được tổ chức thành Năm, Quý, Tháng, Tuần và Ngày. Với sự sắp xếp này, chiều Thời gian không còn phân cấp vì có những tuần trong năm có thể thuộc về nhiều tháng khác nhau.

Vì vậy, nếu mỗi chiều chứa nhiều mức trừu tượng, dữ liệu có thể được xem từ nhiều khung nhìn linh động khác nhau. Một số thao tác điển hình của khối dữ liệu như roll-up (tăng mức độ trừu tượng), drill-down (giảm mức độ trừu tượng hoặc tăng mức chi tiết), slice and dice (chọn và chiếu), và pivot (định hướng lại khung nhìn đa chiều của dữ liệu), cho phép tương tác truy vấn và phân tích dữ liệu rất tiện lợi. Những thao tác đó được biết như Xử lý phân tích trực tuyến (OnLine Analytical Processing – OLAP). Những nhà ra quyết định thường có những câu hỏi có dạng như “tính toán và xếp hạng tổng số lượng hàng hoá bán được theo mỗi quốc gia (hoặc theo mỗi năm)”. Họ cũng muốn so sánh hai độ đo số học như số lượng hàng bán và ngân sách được tổng hợp bởi cùng các chiều. Như vậy, một đặc tính để phân biệt của mô hình dữ liệu đa chiều là nó nhấn mạnh sự tổng hợp của các độ đo bởi một hoặc nhiều chiều, mà đó là một trong những thao tác chính yếu để tăng tốc độ xử lý truy vấn.

* + 1. **Kiến trúc khối (Cube) của OLAP**

Đối tượng chính của OLAP là khối (cube), một thể hiện đa chiều của dữ liệu chi tiết và tổng hợp. Một khối bao gồm một nguồn dữ liệu (Data source), các chiều (Dimensions), các độ đo (Measures) và các phần dành riêng (Partitions). Các khối được thiết kế dựa trên yêu cầu phân tích của người dùng. Một kho dữ liệu có thể hỗ trợ nhiều khối khác nhau như khối Bán hàng, khối Bảng kiểm kê…

Dữ liệu nguồn của một khối chỉ ra nơi chứa kho dữ liệu cung cấp dữ liệu cho khối. Các chiều (dimension) được ánh xạ từ các thông tin của các bảng chiều (dimension table) trong kho dữ liệu vào các mức phân cấp, ví dụ như chiều Địa lý thì gồm các mức như Lục địa, Quốc gia, Tỉnh-Thành phố. Các chiều có thể được tạo một cách độc lập và có thể chia sẻ giữa các khối nhằm xây dựng các khối dễ dàng và để chắc chắn rằng thông tin tổng hợp cho phân tích luôn ổn định. Ví dụ, nếu một chiều chia sẻ một phân cấp sản phẩm và được sử dụng trong tất cả các khối thì cấu tạo của thông tin tổng hợp về sản phẩm sẽ ổn định giữa các khối sử dụng chiều đó.

Một chiều ảo (virtual dimension) là một dạng đặc biệt của chiều mà ánh xạ các thuộc tính từ các thành viên (member) của một chiều khác để sau đó có thể được sử dụng trong các khối. Ví dụ, một chiều ảo của thuộc tính kích thước sản phẩm cho phép một khối (cube) tổng hợp dữ liệu như số lượng sản phẩm bán được theo kích thước, hoặc như số lượng áo bán được theo kiểu và theo kích thước. Các chiều ảo (virtual dimension) và các thuộc tính thành viên được đánh giá là cần thiết cho các truy vấn và chúng không đòi hỏi phải có các khối lưu trữ vật lý.

Các độ đo (measure) xác định các giá trị số từ bảng sự kiện (fact table) mà được tổng hợp cho phân tích như giá bán, chi phí hoặc số lượng bán.

Các phần dành riêng (partition) là các vật chứa lưu trữ đa chiều, giữ dữ liệu của khối. Mỗi khối chứa ít nhất một partition, và dữ liệu của khối có thể kết hợp từ nhiều partition. Mỗi partition có thể lấy dữ liệu một nguồn dữ liệu khác nhau và có thể lưu trong một vị trí riêng biệt (separate).Dữ liệu của một partition có thể được cập nhật độc lập với các partition khác trong một khối. Ví dụ, dữ liệu của một khối có thể được chia theo thời gian, với một partition chứa dữ liệu của năm hiện hành, một partition khác chứa dữ liệu của năm trước, và một partition thứ ba chứa tất cả dữ liệu của các năm trước nữa.

Các partition của một khối có thể được lưu trữ độc lập trong các cách thức khác nhau với các mức độ tổng kết khác nhau. Các partition không thể hiện đối với người dùng, đối với họ một khối (cube) là một đối tượng đơn, và chúng cung cấp các tuỳ chọn đa dạng để quản lý dữ liệu OLAP. Một khối ảo (virtual cube) là một khung nhìn luận lý (logic) của các phần chia của một hoặc nhiều khối.

Một khối ảo có thể được sử dụng để nối (join) các khối khác nhau để chia sẻ một chiều chung nào đó, ví dụ như có thể kết giữa khối Bán hàng và khối Kho nhằm các mục đích phân tích đặc biệt nào đó trong khi duy trì các khối tách biệt cho đơn giản. Các chiều (dimension) và các độ đo (measure) có thể được chọn từ các khối được kết để thể hiện trong khối ảo.

* + 1. **So sánh OLAP và OLTP**

Đặc trưng của các ứng dụng OLTP (On-Line Transaction Processing) là các tác vụ xử lý tự động ghi chép dữ liệu xử lý tác vụ của một tổ chức như ghi nhận đơn đặt hàng và các giao dịch ngân hàng (chúng là những công việc hàng ngày của tổ chức thương mại) mà cần phải đọc hoặc cập nhật một vài mẩu tin dựa trên khoá chính của chúng. Những tác vụ đó có cấu trúc, được lặp lại, bao gồm các giao dịch ngắn, tối giản và tách biệt, yêu cầu dữ liệu chi tiết và mới cập nhật. Các cơ sở dữ liệu tác nghiệp có xu hướng từ vài trăm megabyte đến hàng gigabyte kích thước và chỉ lưu trữ các dữ liệu hiện hành. Tính nhất quán và khả năng phục hồi của cơ sở dữ liệu là then chốt, và tối đa thông lượng giao dịch là thước đo chính yếu. Vì thế cơ sở dữ liệu được thiết kế để tối thiểu các xung đột trùng lặp.

Còn kho dữ liệu, mục tiêu là hỗ trợ quyết định cho các nhà quản lý. Tính chi tiết và riêng lẻ của các mẩu tin thì ít quan trọng hơn tính lịch sử, tổng kết và hợp nhất của dữ liệu. Do đó, kho dữ liệu thường chứa dữ liệu hợp nhất từ một hoặc nhiều cơ sở dữ liệu tác nghiệp và được thu thập qua một thời gian dài. Kết quả là kích thước kho dữ liệu có khuynh hướng từ vài trăm gigabyte đến hàng terabyte so với các cơ sở dữ liệu tác nghiệp. Kho dữ liệu hỗ trợ các truy vấn phức tạp với thời gian hồi đáp nhanh, các truy vấn phức tạp có thể truy xuất hàng triệu mẩu tin và thực hiện nhiều lần các thao tác quét, kết và tổng hợp. Đối với kho dữ liệu, số lượng truy vấn đưa vào và thời gian hồi đáp quan trọng hơn số lượng giao dịch đưa vào. Mà OLAP là một trong những công cụ cho phép thực hiện hiệu quả các truy vấn này.

Căn cứ vào đó, các cơ sở dữ liệu tác nghiệp được xây dựng để hỗ trợ tốt các tác vụ OLTP, vì thế nếu cố gắng thực thi các truy vấn OLAP phức tạp đối với các cơ sở dữ liệu tác nghiệp sẽ cho kết quả là hiệu quả thực hiện không thể chấp nhận được.

* + 1. **Các thành phần của OLAP**

Những thành phần mà OLAP sử dụng để thực hiện các dịch vụ bao gồm:

- Nguồn dữ liệu: Các cơ sở dữ liệu OLTP và các nguồn dữ liệu hợp lệ khác chứa các dữ liệu có thể chuyển đổi thành dữ liệu OLAP trong kho lưu trữ.

- Kho trung gian: là nơi lưu trữ và xử lý dữ liệu được tập hợp, sau đó được sắp xếp, sàng lọc, chuyển đổi thành dữ liệu OLAP hữu ích.

- Máy chủ lưu trữ: Các máy tính chạy cơ sở dữ liệu liên kết chứa các kho dữ liệu cho kho lưu trữ, và các máy chủ quản lý dữ liệu OLAP (warehouse server).

- Ứng dụng thông minh: Các bộ công cụ và ứng dụng thực hiện truy vấn dữ liệu OLAP và cung cấp các báo cáo và thông tin cho người ra quyết định của doanh nghiệp (Business Intelligence).

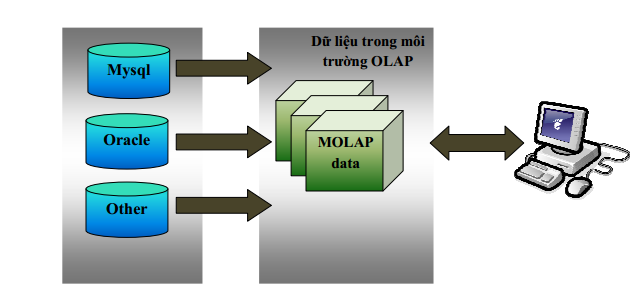
- Siêu dữ liệu: Các đối tượng như các bảng biểu trong cơ sở dữ liệu OLTP, các khối trong kho lưu trữ dữ liệu, và các bản ghi mà ứng dụng tham chiếu tới các đoạn dữ liệu khác nhau.

* + 1. **Các mô hình lưu trữ hỗ trợ OLAP**

Dịch vụ OLAP hỗ trợ nhiều mô hình lưu trữ dữ liệu khác nhau, mỗi mô hình có các ưu và khuyết điểm riêng, chúng được sử dụng tuỳ theo mục đích khai thác.

1. Mô hình Multidimentional OLAP (MOLAP)

Mô hình OLAP đa chiều (MOLAP) lưu trữ dữ liệu cơ sở (là dữ liệu từ các bảng của kho dữ liệu hoặc data mart) và thông tin tổng hợp (là các độ đo được tính toán từ các bảng) trong các cấu trúc đa chiều gọi là các khối (cube). Các cấu trúc này được lưu bên ngoài cơ sở dữ liệu data mart hoặc kho dữ liệu.



Hình 2. 6 Mô hình MOLAP

Lưu trữ các khối (cube) trong cấu trúc MOLAP là tốt nhất cho các truy vấn tổng hợp dữ liệu thường xuyên mà cần thời gian hồi đáp nhanh. Ví dụ, tổng sản phẩm bán được của tất cả các vùng theo quý.

* Ưu điểm của mô hình MOLAP:

- Thực thi nhanh: khối trong MOLAP thu hồi dữ liệu nhanh và tối ưu hóa hoạt động.

- Có thể thực hiện các phép toán phức tạp: mọi tính toán được tạo ra trước khi khối tạo ra.

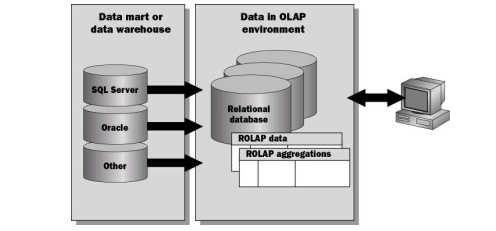
* Nhược điểm của mô hình MOLAP:

- Giới hạn lượng dữ liệu có thể xử lý: Bởi vì tất cả các tính toán được sinh ra khi xây dựng khối, do đó nó không thể bao gồm lượng dữ liệu lớn trong khối của chính nó. Điều này không có nghĩa là dữ liệu từ khối không thể được xây dựng từ một lượng dữ liệu lớn. Điều này có thể, nhưng nó chỉ tóm tắt thông tin chứa trong chính nó.

- Yêu cầu đầu tư thêm: Công nghệ tạo khối thường được độc quyền và không tồn tại trong tổ chức nào. Vì vậy, để sử dụng công nghệ MOLAP cần phải đầu tư bổ sung thêm vốn và nhân lực

1. Mô hình Relational OLAP (ROLAP)

Mô hình OLAP quan hệ (ROLAP) lưu trữ dữ liệu cơ sở và thông tin tổng hợp trong các bảng quan hệ. Các bảng này được lưu trữ trong cùng cơ sở dữ liệu như là các bảng của kho dữ liệu.



Hình 2. 7 Mô hình ROLAP

Lưu trữ các khối trong cấu trúc ROLAP là tốt nhất cho các truy vấn dữ liệu không thường xuyên. Ví dụ như nếu 80% người dùng truy vấn chỉ dữ liệu trong vòng một năm trở lại đây, các dữ liệu cũ hơn một năm sẽ được đưa vào một cấu trúc ROLAP để giảm không gian đĩa bị chiếm dụng, hơn nữa còn để loại trừ dữ liệu trùng lặp.

* Ưu điểm của mô hình ROLAP:

- Có thể xử lý lượng dữ liệu lớn: Kích thước giới hạn của ROLAP phụ thuộc vào kích thước của cơ sở dữ liệu ngồn. Nói cách khác, bản thân công nghệ ROLAP không có giới hạn về kích thước dữ liệu.

- Có thể vận dụng chức năng vốn có của cơ sở dữ liệu quan hệ: Cơ sở dữ liệu quan hệ thường đi kèm với rất nhiều chức năng. Công nghệ ROLAP có thể tận dụng các chức năng này, tiết kiệm chi phí.

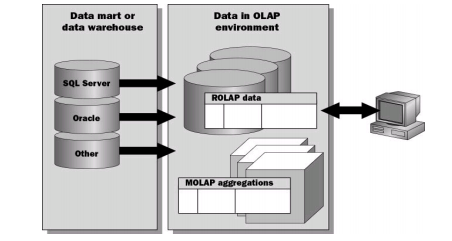
* Nhược điểm của ROLAP:

- Hiệu suất xử lý thấp: Mỗi báo cáo ROLAP thường được tập hợp dữ liệu từ nhiều bảng khác nhau, điều này sẽ làm cho hiệu quả của ROLAP thấp khi dữ liệu lớn, phân tán.

- Giới hạn bởi chức năng của SQL: Bởi vì công nghệ ROLAP chủ yếu dựa vào việc tạo ra các câu lệnh SQL để truy vấn cơ sở dữ liệu. Mà báo cáo dựa trên truy vấn SQL trong một số trường hợp không đạt được hiệu quả mong muốn. Các nhà phát triển đã khắc phục điều này bằng các tạo ra các công cụ hỗ trợ ngoài giúp người dùng tạo ra các chức năng của riêng họ

1. Mô hình Hybird OLAP (HOLAP)

Mô hình OLAP lai (HOLAP) là sựkết hợp giữa MOLAP và ROLAP.



Hình 2. 8 Mô hình HOLAP

Lưu trữ các khối (cube) trong cấu trúc HOLAP là tốt nhất cho các truy vấn tổng hợp dữ liệu thường xuyên dựa trên một lượng lớn dữ liệu cơ sở. Ví dụ, chúng ta sẽ lưu trữ dữ liệu bán hàng theo hàng quý, hàng năm trong cấu trong MOLAP và dữ liệu hàng tháng, hàng tuần và hàng ngày trong cấu trúc ROLAP.

* Lợi ích của việc lưu trữ trong cấu trúc HOLAP là:

- Lấy dữ liệu trong khối (cube) nhanh hơn bằng cách sử dụng xử lý truy vấn tốc độ cao của MOLAP.

- Tiêu thụ ít không gian lưu trữ hơn MOLAP.

- Tránh trùng lặp dữ liệu.

* + 1. **So sánh các mô hình OLAP**

|  |  |  |  |
| --- | --- | --- | --- |
|  | MOLAP | ROLAP | HOLAP |
| Lưu trữ dữ liệu cơ sở | Khối | Bảng quan hệ | Bảng quan hệ |
| Lưu trữ thông tin tổng hợp | Khối | Bảng quan hệ | Khối |
| Hiệu suất thực hiện truy vấn | Nhanh nhất | Chậm nhất | Nhanh |
| Tiêu thụkhông gian lưu trữ | Nhiều | Thấp | Trung bình |
| Chi phí bảo trì | Cao | Thấp | Trung bình |

* 1. **Giới thiệu bộ công cụ Pentaho**
  2. **Giời thiệu về Ofbiz**
     1. **Giới thiệu Framework Ofbiz**

OFbiz là một mã nguồn mở cung cấp framework cho việc phát triển một ứng

dụng web trên nền Java. OFbiz cung cấp sẵn các thành phần cần thiết cho ứng dụng.

Phát triển ứng dụng dựa trên OFbiz là việc viết mã sử dụng các thành phần có sẵn để phối hợp chúng hoạt động thành một ứng dụng hoàn chỉnh. Công việc xây dựng một ứng dụng web trên OFbiz cũng tương tự xây một ngôi nhà, ngôi nhà đã có sẵn nền móng và các thành phần khác như tường, cửa, v.v… Công việc của người thợ là

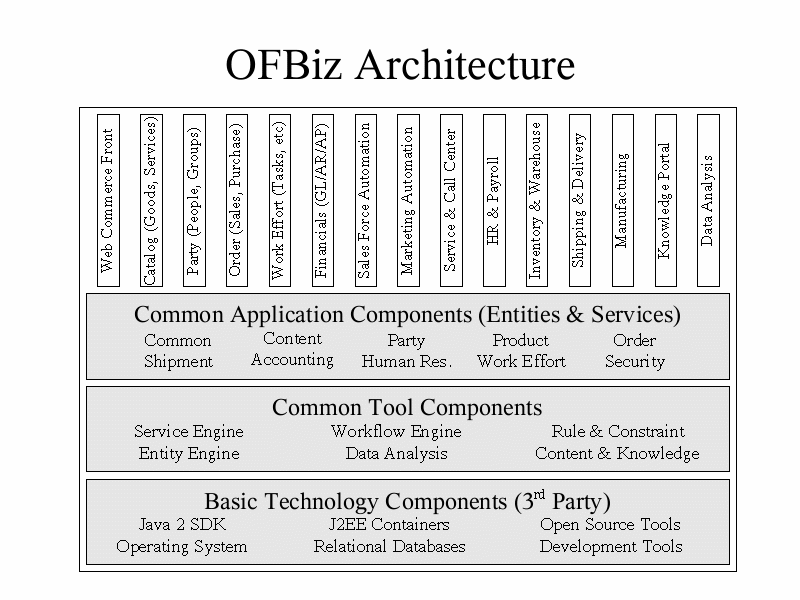
kết hợp các bức tường theo một cách nào đó và đặt các ô cửa ở các vị trí thích hợp.

OFBiz là framework mã nguồn mở được thiết kế để tạo thuận lợi cho việc xây dựng ERP software (ERP là tên chung cho bất kì hệ thống nào mà tích hợp tất cả quy trình business và dữ liệu thành 1 hệ thống duy nhất).



Hình 2. 9 Các thành phần ERP

Kiến trúc OFbiz gồm 04 lớp: Công nghệ cơ sở (Basic technology), Công cụ chung (Common tool), ứng dụng chung (Common application) và ứng dụng (Application)



Hình 2. 10 Kiến trúc Ofbiz

* Lớp công nghệ cơ sở cung cấp các công nghệ mới nhất để phát triển phần mềm trên J2EE, OFbiz hỗ trợ JavaBean, CORBA, Web Service, các cơ sở dữ liệu quan hệ, các công cụ để phát triển và quản trị hệ thống. Lớp thứ nhất của OFbiz là chức năng tương tự như một Application Server phục vụ các chức năng cơ bản của máy chủ dữ liệu.
* Lớp công cụ chung cung cấp các dịch vụ hỗ trợ cho việc phát triển phần mềm. Entity Engine cung cấp một cách thức dễ dàng để truy xuất vào cơ sở dữ liệu, không cần phải viết các câu lệnh SQL mà chỉ cần dùng các phương thức của Entity Engine để truy xuất và thao tác dữ liệu. Entity Engine thay thế cho việc dùng JDBC và các lệnh SQL chuẩn. Service Engine cung cấp cho lập trình viên thao tác lập trình trên mô hình MVC được dễ dàng và thuận lợi. Workflow engine dựa trên đặc tả WFMC và OMG. Nó là một phần của Services framework và được tích hợp chặt chẽ với Entity Engine, OFbiz Workflow Engine sử dụng XPDL làm ngôn ngữ định nghĩa tiến trình (process definition language). OFbiz Rule Engine dựa trên kỹ thuật lập trình logic đã tồn tại hàng thập kỉ. Ngôn ngữ lập trình logic được nhiều người biết đến là Prolog, mặc dù có nhiều ngôn ngữ khác nữa, ý tưởng chính ở đây là thay vì đưa ra một tập các chỉ thị thực thi theo một trình tự cố định như các ngôn ngữ truyền thống, chúng ta chỉ cần định nghĩa các sự kiện (fact) và các luật (rule).
* Lớp ứng dụng chung là các ứng dụng cơ bản nhất đối một doanh nghiệp, OFbiz đã xây dựng các ứng dụng này thành các thành phần cơ bản, khi phát triển ứng dụng các nhà phát triển phần mềm chỉ khảo sát thực tế, tích hợp các thành phần ứng dụng này lại theo đúng yêu cầu khảo sát là có thể sử dụng được. Việc này giúp cho việc phát triển phần mềm trở nên nhanh chóng và dễ nâng cấp, bảo trì.
* Lớp ứng dụng trên cùng là các ứng dụng chuẩn của doanh nghiệp, các doanh nghiệp chỉ cần chỉnh lại cấu hình để đáp ứng đúng thực tế của doanh nghiệp. Các ứng dụng này hầu như đã đáp ứng được hết các yêu cầu của doanh nghiệp.

Như đã phân tích ở trên, kiến trúc OFbiz là một flatform, một framework để cho việc phát triển một ứng dụng trên web được nhanh chóng và dễ dàng. Tùy theo yêu cầu cụ thể của doanh nghiệp, nếu sát với lớp trên cùng thì sử dụng ngay lớp đó, nếu không thì phải hiệu chỉnh lại lớp ứng dụng, nếu vẫn không sát thực tế thì phải phát triển ứng dụng trên lớp công cụ chung, nếu vẫn còn chưa được thì nhà phát triển phần mềm phải sử dụng lớp công nghệ cơ sở để phát triển ứng dụng. Càng về lớp trên thời gian phát triển càng nhanh nhưng độ linh hoạt càng thấp, lớp dưới cùng thì thời gian phát triển ứng dụng lâu hơn nhưng giải quyết được nhiều việc hơn.

Các thành phần trong Ofbiz bao gồm :

* Accounting (Agreements, Invoicing, Vendor Management, General Ledger ): Phần mềm kế toán cho công ty
* Asset Maintenance: Bảo vệ tài sản cho nhân viên trong công ty
* Catalogue and Product Management: Quản lý sản phẩm và calatogue
* Facility and Warehouse Management: Quản lý kho và cơ sở vật chất
* Manufacturing: Quản lý sản xuất của công ty
* Order Processing: Xử lý thông tin đặt hàng của khách hàng
* Inventory Management, automated stock replenishment: Kiểm toán, tự động bổ sung cổ phiếu
* Content management system (CMS): Quản lý nội dung hệ thống
* Human resources (HR): Quản trị nhân sự trong công ty
* People and Group Management: Quản lý cá nhân và nhóm người lao động
* Project Management: Quản lý dự án
* Sales Force Automation: Bán hàng tự động
* Work Effort Management: Quản lý kết quả làm việc
* Electronic point of sale (ePOS): Quản lý chi nhánh báng hàng điện tử
* Electronic commerce (eCommerce): Thương mại điện tử
  + 1. **Kiến trúc 3 tầng và cách thức phát triển ứng dụng với Ofbiz**

1. ***Kiến trúc 3 tầng của Ofbiz***

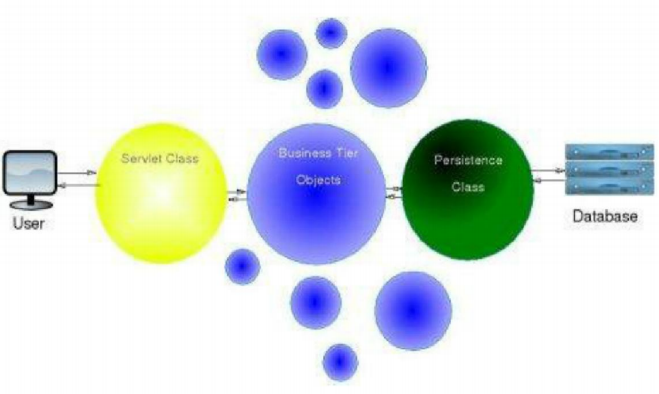
Framework Ofbiz sử dụng mô hình 3 tầng chung cho tất cả các ứng dụng của nó. Đó là tầng Data Layer, Business Logic Layer và Presentation Layer. Các tầng Data Layer và Service Layer có những cách thức riêng chịu trách nhiệm tương tác với từng tầng

* **Data Model Layer**: Tầng này đại diện cho Cơ sở dữ liệu. Nó bao gồm 1 thực thể chức năng (Entity Engine) có trách nhiệm kết nối cơ sở dữ liệu, phục hồi cơ sở dữ liệu và lưu trữ… Nó sử dụng lớp Java Generic Delegator để kết nối với cơ sở dữ liệu và nó sử dụng Java Generic Value đại diện cho những thực thể được chèn vào cơ sở dữ liệu.
* **Business Logic Layer**: Tầng này đại diện cho logic, hoặc các dịch vụ cung cấp cho người dùng và thực hiện trên Cơ sở dữ liệu của tầng Data layer. Có thể có nhiều loại dịch vụ như java, SOAP, simple, workflow… và mỗi lại dịch vụ có cách thức riêng của mình. Có một Service Engine có trách nhiệm phân chia và gọi các dịch vụ..v.v..
* **Presentation Layer**: Ofbiz đã chuyển sang sử dung “Screens” để đại diện cho từng trang của Ofbiz. Do đó mỗi trang nên được biểu diễn bình thường như một screen. Một trang Ofbiz bao gồm nhiều thành phần như hearders, Footers, appheader… do đó khi dựng hình các trang, đây là sự kết hợp có trình tự đã được sắp đặt, hoặc bao gồm, trong một screen

1. ***Phát triển ứng dụng với OfBiz – Overview***

So sánh việc phát triển bằng Ofbiz với các ứng dụng truyền thống : Java, PHP/Perl:

* Java:



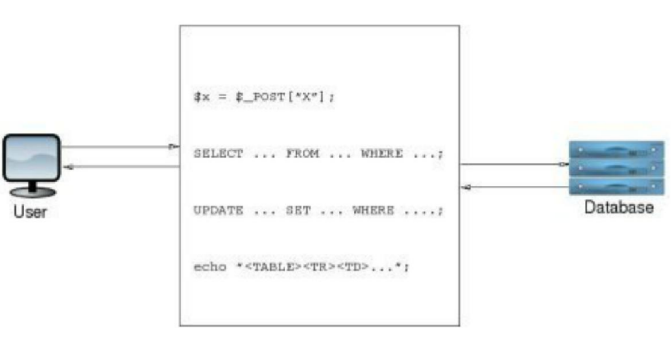
Hình 2. 11 Phát triển ứng dụng Java

Quá trình phát triển Java làm nổi bật sự tách biệt rõ ràng hoặc các tầng của code, thích hợp cho những nhóm phát triển lớn với chức năng riêng biệt của từng đội. Có các tầng riêng biệt cho việc xử lí như tầng business logic, managing databases và hiển thị kết quả cho người dùng.

Dữ liệu là những mô hình – giống như những đối tượng, với thành phần cho các trường và nhận được, thiết lập những mô hình cho việc truy cập vào các trường. Những đối tượng vẫn “Tiếp tục tồn tại”, hoặc được lưu trữ vào 1 cơ sở dữ liệu quan hệ, bằng cách sử dụng 1 framework persistence.

Điển hình ứng dụng Java bao gồm nhiều thư viện, classes, property files, các files định nghĩa, và các files triển khai. Chính code của nó đã thành lập các class riêng biệt, trong mỗi class đó là những tập tin riêng của chính nó, và các files class được nhóm thành các gói trong thư mục riêng biệt.

* PHP/Perl:



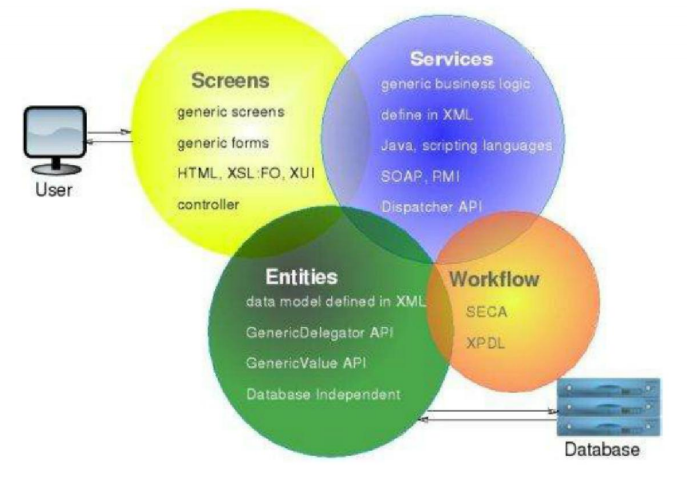
Hình 2. 12 Phát triển ứng dụng php

PHP và Perl, thì ngược lại, nổi bật là sự phát triển nhanh và kết quả ngay lập tức, có thể do 1 nhóm nhỏ các nhà phát triển. Hầu hết code của Perl và PHP là đơn tầng. Ví dụ, 1 trang đơn hoặc mã script có cấu trúc từ input, tác động qua lại với cơ sở dữ liệu, và hoàn lại output.

Mặc dù cả PHP và Perl là ngôn ngữ hướng đối tượng, dữ liệu thường được sử dụng với danh sách và bản đồ ( list and maps ), chứ ko phải đối tượng công khai. Cơ sở dữ liệu truy cập được xử lý bằng câu lệnh SQL trong trang và Scripts.

PHP và Perl là ngôn ngữ script – cái mà được biên dịch trong thời gian chạy, do đó gia tăng thử nghiệm và phát triển.

* OFBIZ:



Hình 2. 13 Phát triển ứng dụng Ofbiz

Ofbiz kết hợp giữa tốc độ của PHP/Perl với cấu trúc của Java. Mục đích quan trọng nhất của Ofbiz là giảm số lượng mã yêu cầu. Điều này đạt được bằng cách cho phép các nhà phát triển tạo ra những định nghĩa cao cấp của data model, business logic, presentation và sau đó cung cấp với 1 bộ API chung có nhiệm vụ làm công việc định nghĩa.

Ví dụ, nhiệm vụ chung của ứng dụng kinh doanh là tạo ra các bảng để lưu trữ dữ liệu và viết các forms để update các bảng ấy. Điều này thường yêu cầu mã HTML với nhiều lần lặp đi lặp lại của các bảng cơ sử dữ liệu. Với OFBiz, thực thể và form widget cho phép form đó được tạo 1 cách tự động dựa trên các định nghĩa meta.

Giống như Java, OFBiz là tầng cao và mang tính hệ thống. Nhưng phần của Framework Ofbiz và ứng dụng Ofbiz nằm trong thành phần riệng biệt và Ofbiz cho phép các ứng dụng bên ngoài hoặc các ứng dụng tùy chỉnh được thêm vào 1 cách dễ dàng. Mỗi ứng dụng được chia thành view (presentation), business logic và tầng data model.

Tầng view của Ofbiz được xây dựng quanh mô hình Framework MVC và được trang trí bên ngoài. Trong PHP hoặc CGI Perl, người truy cập trang web yêu cầu cụ thể từng trang hoặc scripts, cái mà tập hợp dữ liệu và tạo ra toàn bộ trang và trả lại cho người truy cập. Trong Ofbiz, ngược lại, điều khiển phân tích cú pháp là yêu cầu đầu tiên, sau đó mới xác định những việc cần làm. Tầng view là sự tách xa hơn trong presentation ( ví dụ HTML hoặc XSL:FO) và view logic( ví dụ thu thập dữ liệu để tạo thành 1 form.) Tài liệu được trình bày là tự được tách thành từng phần, giống như header, footer, leftbar, rightbar và ứng dụng header, cái được tạo ra cùng nhau trong “trang trí” thời trang.

Sự thuận lợi của MVC và các mẫu trang trí trở lên rõ ràng cho các ứng dụng rộng hơn, nơi những hành động tượng tự và các yếu tố được lặp lại nhiều lần. OFBiz cho phép sự tái sử dụng các các yếu tố thiết kế , view logic và form action, giảm bớt sự cần thiết cho việc tạo và, thậm chí tệ hơn, duy trì code lặp đi lặp lại.

Tầng business logic của ứng dụng OFBiz không ràng buộc với những trang cụ thể (giống như trong PHP), các đối tượng cụ thể (giống như trong Java), hoặc các server yêu cầu cụ thể (giống như trong CGI Perl hoặc Java servlets.) Thay vào đó chúng thường được tạo ra như những “dịch vụ” nhỏ, những phần code nhỏ có thể dùng lại là những phần có thể viết trong những trạng thái khác nhau của ngôn ngữ, bao gồm Java, Jython, beanshell, hoặc Ofbiz minalang. Dịch vụ sau đó có thể gọi trực tiếp từ web ứng dụng, với những form phân tích tự động, từ trong các dịch vụ khác, từ các workflows công khai hoặc các mức rất nhỏ như SOAP hoặc RMI.

Tầng data access, được xử lý với các thực thể trong OFBiz, cái mà cho phép dữ liệu được mô phỏng trong tập tin XML và truy cập với đối tượng duy nhất GenericDelegator. Trường hợp các mục dữ liệu được xử lý như GenericEntity và đối tượng GenericValue chứ không phải là đối tượng của riêng mình. Như vậy, không cần phải viết (hoặc tạo) cài đặt và thiết lập mã persistence riêng. Thay vào đó, làm việc với dữ liệu trong OFBiz tương đối giống làm việc với danh sách và bản đồ trong Perl hoặc PHP. Thậm chí lưu trữ dữ liệu còn đơn giản hơn. Không cần viết thêm câu lệnh SQL INSERT hoặc UPDATE , chỉ cần sử dụng 1 phương thức lưu trữ().

Bởi vì hầu hết lớp view và nhiều (mặc dù không phải tất cả) của tầng business logic được viết ra bởi ngôn ngữ scripting giống như beanshell hoặc minilang, Framework OFBiz cho phép gia tăng thử nghiệm và phát triển 1 style giống như Perl hoặc PHP.

Cuối cùng, do OFBiz trình bày một ngăn xếp tích hợp từ tầng data tới tầng business logic tới lớp layer, nó có thể được nhận biết qua lại, Ví dụ, thay đổi tầng data có thể tự động thay đổi dịch vụ (business logic) và forms(UI) đang làm việc với dữ liệu. Điều này có thể làm giảm đáng kể số tiền bảo dưỡng cho ứng dụng.

* + 1. **Tìm hiểu thành phần và chạy ứng dụng sẵn có trên OFBIZ**

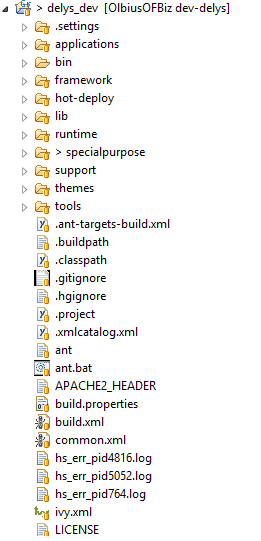
1. ***Các thành phần chính của OFBIZ***

Download Ofbiz tại đây: http://ofbiz.apache.org/download.html

Sau khi download và giải nén chúng ta sẽ có một folder OFBiz. Trước tiên chúng ta xem các thành phần trong thư mục này.

Trong đó nội dung từng thành phần như sau:

* application folder: Bao gồm các ứng dụng có trong OFBiz, khi chúng ta tự tạo một ứng dụng riêng (Chứa core application của OFBIZ). Đại diện cho những application liên quan tới business của OFBiz. Có thể tìm thấy những manufacturing, content management, order management …. trong thư mục này.
* framework folder: Bao gồm các thành phần của framework OFBiz, giống như Entity Engine, Service Engine, thư mục chứa các file chung cho tất cả mọi ứng dụng.
* hot-deploy folder: Trong thư mục này có thể có một số folder, nơi các thành phần của ứng dụng được tải tự động mà không cần tải chúng một cách rõ ràng như chúng ta vẫn thấy ở một số ứng dụng khác đặt trong Application folder.
* specialpurpose folder: Bao gồm một số ứng dụng bổ sung như “ecommerce” và “workflow”, tuy nhiên chúng không phải một phần của OFBiz core. Các component do người dùng phát triển nên đặt ở đây.
* themes folder: Bao gồm các file có chức năng thay đổi giao diện người.



Hình 2. 14 Tổ chức thư mục Ofbiz

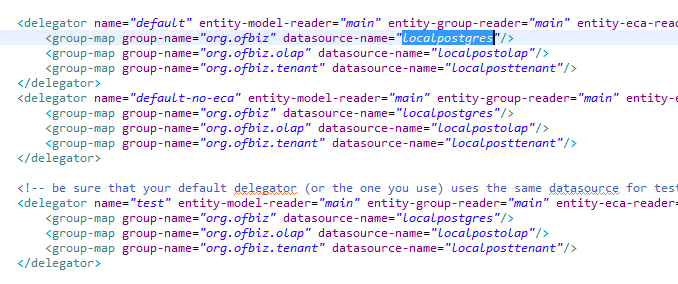
Những thư mục như application, framework, specialpurpose, hot-deploy, themes phải được khai báo trong /framework/base/config/componentload.xml.

1. ***Cấu trúc của một component OFBIZ***

Cấu trúc của 1 component OFBiz điển hình gồm có:

* configuration files (config): Thường được sử dụng cho data configuaration, ví dụ nó được dùng để hỗ trợ các giao diện ngôn ngữ, bên trong nó chúng ta sẽ tìm được một số file cho các ngôn ngữ khác nhau, và dựa trên ngôn ngữ giao diện người dùng, một trong số các file này sẽ được sử dụng.
* seed data definition (data): Chứa đựng dữ liệu load về khi OFBiz được khởi động và dữ liệu demo.
* entity definition (entitydef): Chứa việc thực thi lớp dữ liệu, ví dụ bản quan hệ CSDL và các mối quan hệ. Bên trong thư mục này có 2 file chính, 1 cho việc định nghĩa và 1 cho việc thực thi : entitygroup.xml và entitymodel.xml.
* flow logic (script): Chứa đựng việc thực thi cho các service sử dụng OFBiz mini-language và nó chứa đựng một số script. Bên trong thư mục này còn chứa việc thực thi các service cho các ứng dụng khác nhau.
* service definition (servicedef): Định nghĩa các service được dùng trong lớp business logic, nó chứa các file service.xml – định nghĩa services, service\_view.xml – dùng cho view (Khi add 1 file service, cần phải được định nghĩa thêm trong ofbiz component.xml).
* java code (src): Chứa các file source cho các service – các file được thực thi với Java.
* webapp: Chứa các form và ứng dụng web. Với OFBiz các trang được chia thành các thành phần nhỏ hơn, những cái được kết hợp để tạo lên sản phẩm mới. Do đó nhiều trang có thể chia sẻ các thành phần chung như header, sidebar, và navigation bars.
* widget: Là lớp trình diễn OFBiz gần nhất, các trang được định nghĩa như “screen”. Thư mục widgets cho giao diện màn hình người dùng. OFBiz cho phép thiết kế giao diện người dùng để tạo như “generic screens” có thể được sử dụng cho những nền khác. Nội dung thư mục widget phản ánh những thứ trên webapp. Do đó mỗi ứng dụng sẽ có màn hình riêng của nó. Screen được chia làm 2 phần: action và widget. Action chịu trách nhiệm cho việc lấy dữ liệu về trong khi widget chịu trách nhiệm hiển thị dữ liệu.
  + 1. **Cấu hình để chạy Ofbiz trên localhost**

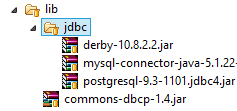
1. Sau khi đã tải thành công ofbiz về máy, ta cần phải tạo cơ sở dữ liệu cho ofbiz để có thể hoạt động:

* Bước 1: Tạo cơ sở dữ liệu hoạt động cho ofbiz và khai báo tại đường dẫn: ofbiz/framework/entity/config/entityengine.xm. 



Hình 2. 15 Cấu hình CDSL trong Ofbiz

Ofbiz sử dụng jdbc để kết nối với cơ sở dữ liệu, vì vậy khi làm việc với hệ quản trị cơ sở dữ liệu nào thi cần phải import thư viện jdbc của thư viện đó tại: ofbiz/framework/entity/lib/jdbc:



Hình 2. 16 Lưu trữ driver jdbc trong Ofbiz

* Bước 2: dùng lệnh trên cửa sổ command line để build project cho lần đầu chạy hoặc có bất cứ thay đổi nào đối với các file java trong ofbiz đều phải thực hiện lệnh build để thực hiện việc thay đổi:
  + Từ của sổ command line, di chuyển đến thư mục chứa ofbiz: *cd ofbiz*
  + Sử dụng lệnh: *ant build*
* Bước 3: Chạy ofbiz:
  + Sử dụng lệnh: ant start
  + Lúc này có thể truy cập vào ofbiz trên đường trình duyệt với đường dẫn: <https://localhost:8080/control/main>

b) Tạo mới một component trong ofbiz

- Từ của sổ command line, di chuyển đến thư mục chứa ofbiz: *cd ofbiz*

- Sử dụng lệnh: ./ant create-component. Sau đó điền các thông số

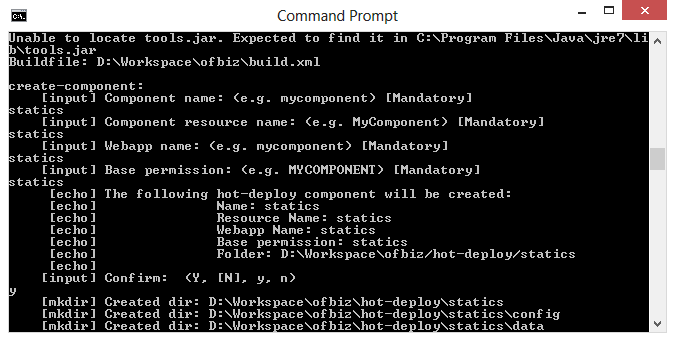
Component name: **statics (***Tên component***)**

Component resource name: **statics** ()

Webapp name: statics (**Tên đường dẫn sẽ được sử dụng để truy cập đến component: https://locahost:8080/hwm**)

Base permission: STM (Quyền được truy cập đến component)

Confirm: Y



Hình 2. 17 Tạo component trong Ofbiz

* Sau khi tạo thành công, component sẽ được tạo tại đường dẫn: ofbiz/ hot-deploy.

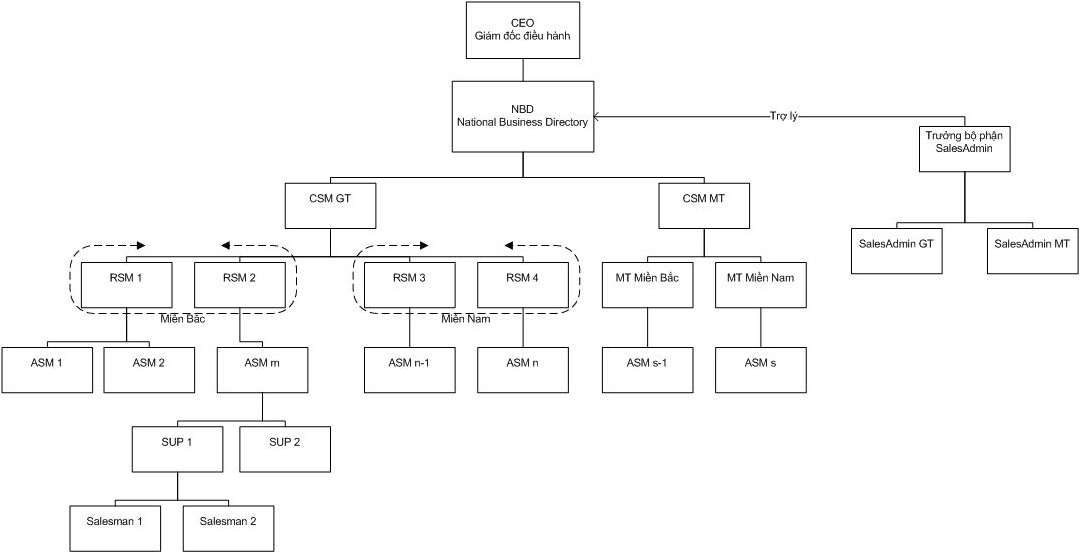
**Chương 3: Mô tả bài toán**

Sau khi đã tìm hiểu các khái niệm về data warehouse và ofbiz ở các chương đầu, phần tiếp theo sẽ được giành để nói về những công việc thực tế mà em đã làm được trong quá trình làm việc và tham gia dự án tại công ty Olbius với đối tác Delys. Đó là việc xây dựng data warehouse và báo cáo cho chủ đề “Bán hàng tại Delys”.

* 1. **Tổ chức nội bộ trong công ty Delys**

Công ty Delys chuyên nhập khẩu và phân phối thực phẩm cao cấp của châu Âu, đặc biệt là các sản phẩm về sữa như váng sữa và sữa chua nhập khẩu. Sản phẩm váng sữa nổi tiếng nhất Việt Nam với tên gọi Monte được công ty nhập khẩu và phân phối độc quyền từ đối tác Zott GMBH & Co. KG (CHLB Đức) – doanh nghiệp sản xuất sữa hàng đầu châu Âu với bề dầy lịch sử từ năm 1926.

1. Tổ chức nội bộ

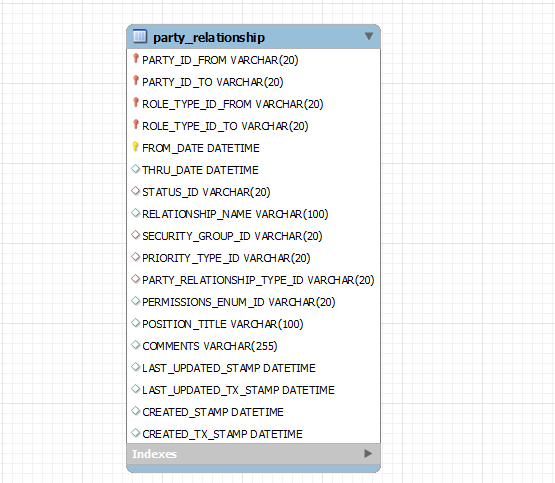


Hình 3. 1 Tổ chức nội bộ Delys

1. Lưu trữ tổ chức của Delys trong Ofbiz

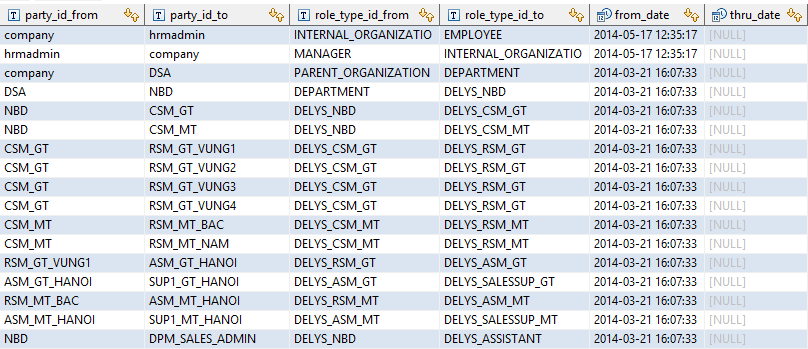
Các mối quan hệ giữa các phòng ban được lưu trong bảng PartyRelationship của Ofbiz.

* Cấu trúc bảng:



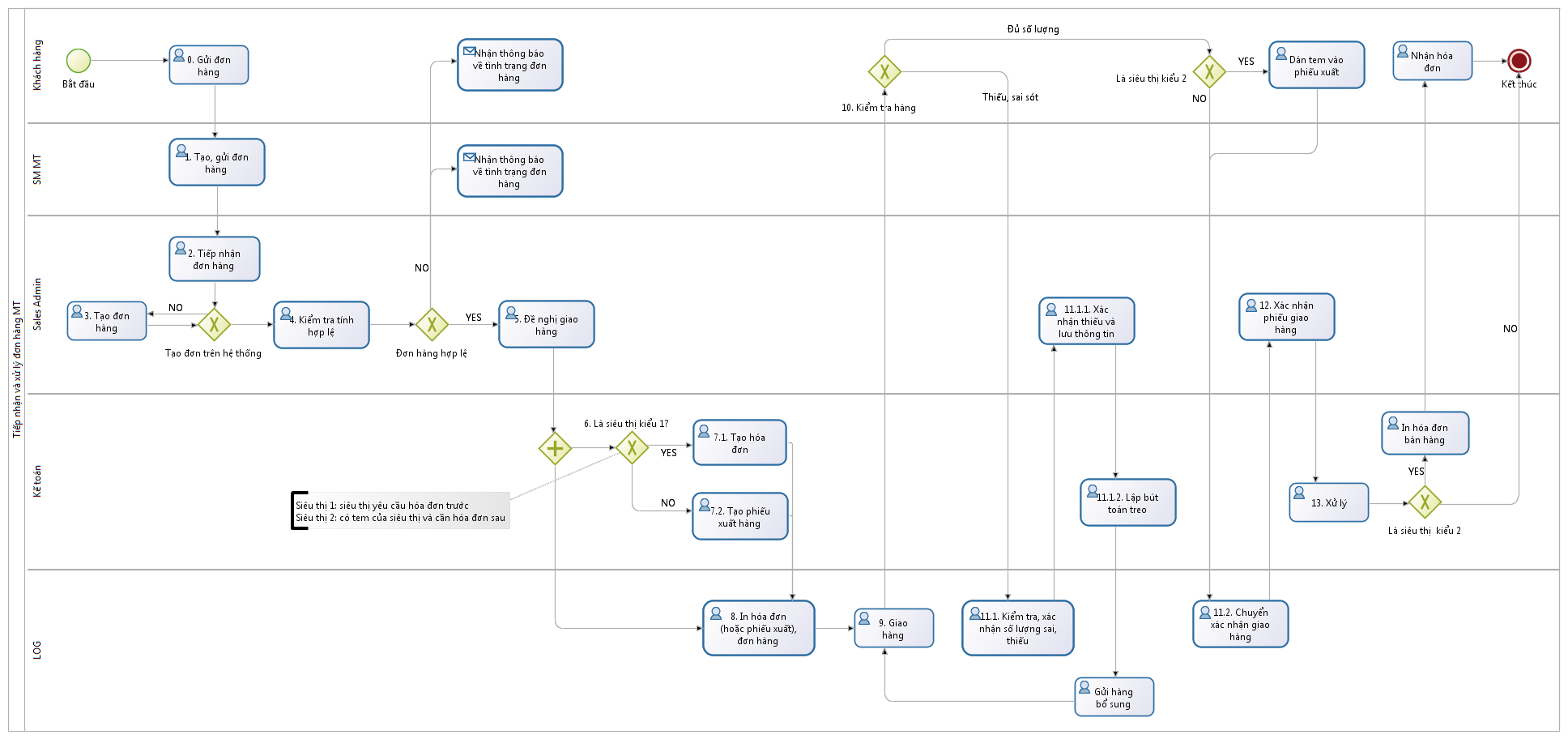
Hình 3. 2 Cấu trúc bảng Party Relationship trong Ofbiz

* Dữ liệu thực tế:



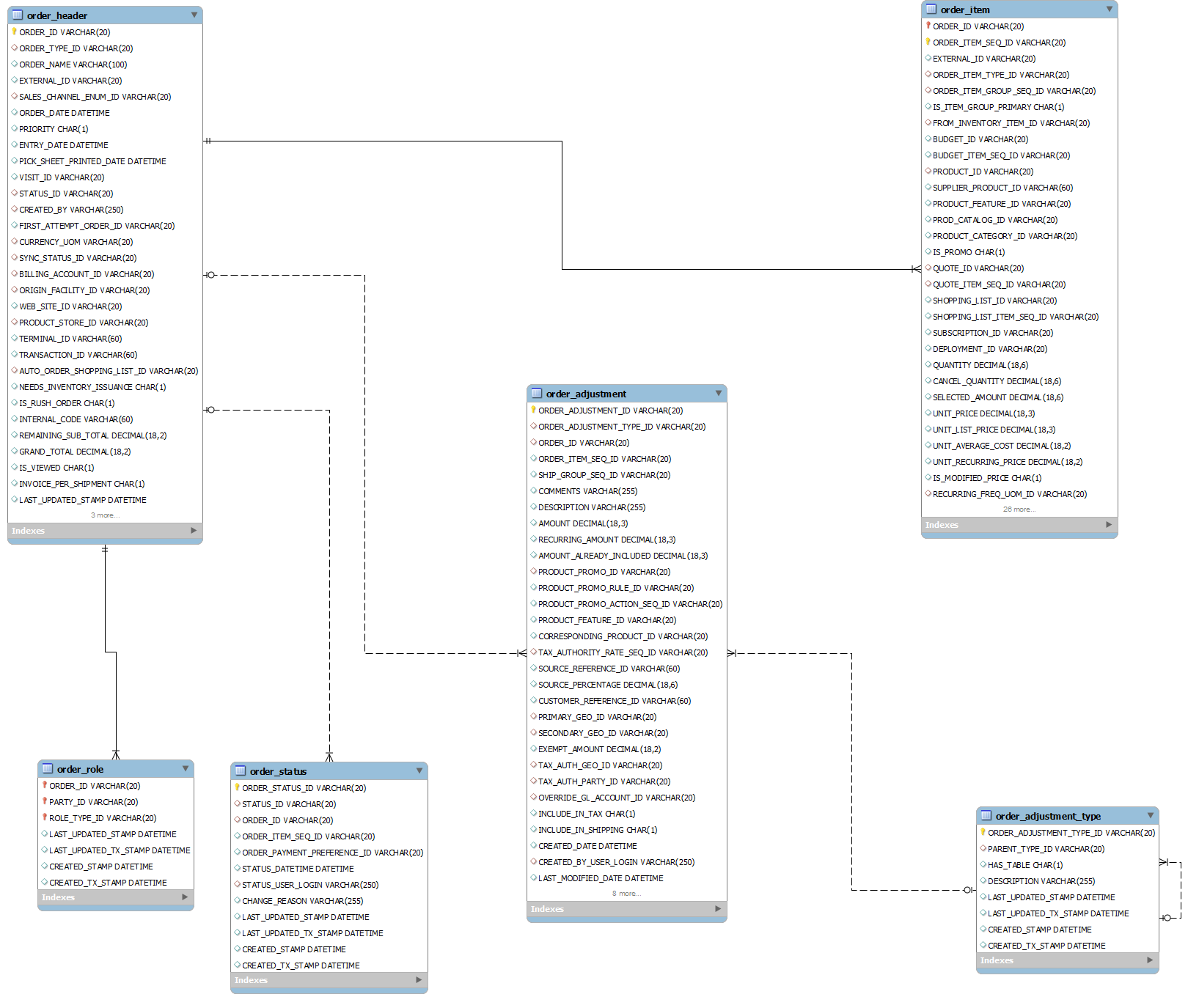
Hình 3. 3 Dữ liệu thực tế Delys trong Party Relationship

* 1. **Quy trình xử lý đơn hàng**
* Quy trình thực hiện tại công ty Delys:



Hình 3. 4 Quy trình xử lý đơn hàng tại Delys

* Dữ liệu được lưu trữ trong Ofbiz:
* Order Header: Lưu các thông tin về các Order đã được tạo trên hệ thống (Kênh bán hàng, giá trị đơn hàng, trạng thái đơn hàng, người tạo đơn hàng…)
* Order Item: Lưu trữ chi tiết các sản phẩm trong đơn hàng
* Order status: Lưu trữ trạng thái, thời gian của Order, Item trong Order theo thời gian. Từ khi tạo đơn hàng đến khi kết thúc. (Thành công hoặc bị hủy)
* Order Adjustment:Lưu trữ sự thay đổi giá trị của đơn hàng so với giá trị ban đầu.Ví dụ:Một đơn hàng có tổng giá trị sản phẩm là 500.000 VNĐ, Khi tính tiền đơn hàng sẽ phải chịu thêm một khoản thuế VAT là 10% thì giá trị này sẽ được lưu tại bảng Order Adjustment
* Order Role: Lưu thông tin các cá nhân, tổ chức liên quan đến đơn hàng. Ví dụ: người đặt hàng, người thanh toán, người tạo hóa đơn …

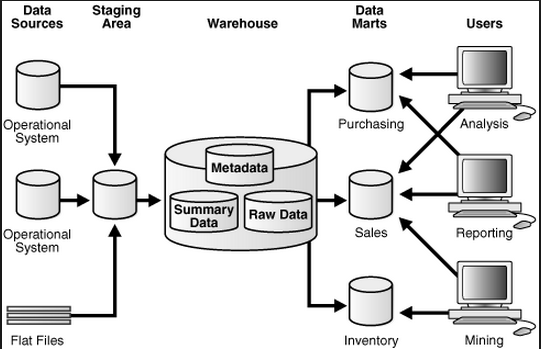


Hình 3. 5 Quan hệ các bảng trong Ofbiz

* 1. **Mô hình xây dựng Data Warehouse cho nghiệp vụ bán hàng**
     1. **Tổng quát**

Như chúng ta đã tìm hiểu , mô hình tổng quát để xây dựng một data warehouse bất kì gồm có ba vùng chính là: data source, staging area( vùng dữ liệu được dùng cho xử lys) và data warehouse.

Ở Delys, dữ liệu từ các công ty con đã được tập hợp vào kho dữ liệu tập trung và ở dưới dạng dữ liệu quan hệ, vì vậy đây là một thuận lợi rất lớn cho những người xây dựng data warehouse.

****

Hình 3. 6 Mô hình data warehouse

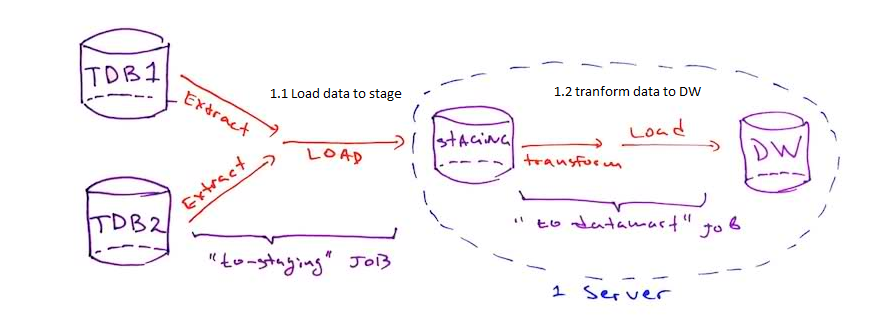
Từ kho dữ liệu tập trung tại công ty, các bảng dữ liệu cần thiết cho việc xây dựng data warehouse sẽ được trích ra và đưa vào vùng staging area để xử lý.

Tại vùng staging area, dữ liệu sẽ được làm sạch và chuyển dạng cho phù hợp với dạng dữ liệu của data warehouse, đây cũng là nơi kiểm tra tính toàn vẹn của dữ liệu trước khi tải vào data warehouse.

Dữ liệu tại staging area sau khi được xử lý sẽ được tải vào data warehouse vào những khoảng thời gian hợp lí.

* + 1. **Mô tả chi tiết**

Sau đây chúng ta sẽ xem xét mô hình chi tiết khi xét cụ thể quá trình xử lý dữ liệu trong staging area và các tiến trình cần có :



Hình 3. 7 Quá trình xử lý dữ liệu trong các vùng DW

Quá trình tổng hợp dữ liệu từ data source vào data warehouse gồm 2 pha :

* Pha 1 : tổng hợp dữ liệu hiện có trong data source vào data warehouse.
* Pha 2 : cập nhật các dữ liệu mới vào data warehouse kể từ thời điểm sau pha 1.

**Trước tiên ta xét pha 1.**

**a) Data source :** đây là kho dữ liệu tập trung của công ty Delys được tổng hợp và cập nhật thường xuyên từ tất cả các phòng ban, nhà phân phối, salesman. Dữ liệu được lưu dưới dạng dữ liệu quan hệ.

**b) Stage :** đây là phần dữ liệu cần thiết cho data warehouse chủ đề “Bán hàng tại Delys” được trích từ data source. Dữ liệu phần này cũng được lưu dưới dạng dữ liệu quan hệ.

Phần này gồm các bảng có cấu trúc gần giống các bảng được trích dữ liệu trong phần data source, ta chỉ loại bỏ các trường mà ta không cần thông tin để tổng hợp vào data warehouse.

**c) Tiến trình 1.1 (Load data to Stage) :** đây là tiến trình tải dữ liệu từ data source vào stage . Thực chất nhiệm vụ của tiến trình này là lọc dữ liệu, loại bỏ những dữ liệu không hợp lệ.

Ví dụ: Đối với trạng thái của các order item, ta chỉ lấy các item có trạng thái là Approve hoặc complete, bỏ những trạng thái khác như: Cancel.

**d)Tiến trình 1.2 (Tranform data to Data Warehouse):** Đây là tiến trình tính toán, chuẩn hóa, dọn dẹp dữ liệu từ Stage trước khi đưa vào lưu trữ tại Data Warehouse.

**Tiếp theo chóng ta sẽ xét đến pha 2.**

Khi đã xây dựng được pha 1 thì pha 2 cũng không phải là vấn đề phức tạp nữa. Thực chất thì pha 2 cũng gần giống như pha 1, nhưng nó cần được tách ra vì pha 2 được thực hiện thường xuyên còn pha 1 chỉ thực hiện 1 lần.

Pha 2 được thực hiện khi có sự thay đổi (Update, tạo mới…) trên cơ sở dữ liệu. Các dữ liệu mới được sinh ra sẽ thực hiện các tiến trình giống như pha 1 để đưa dữ liệu vào Data Warehouse



Hình 3. 8 Cập nhật dữ liệu vào DW

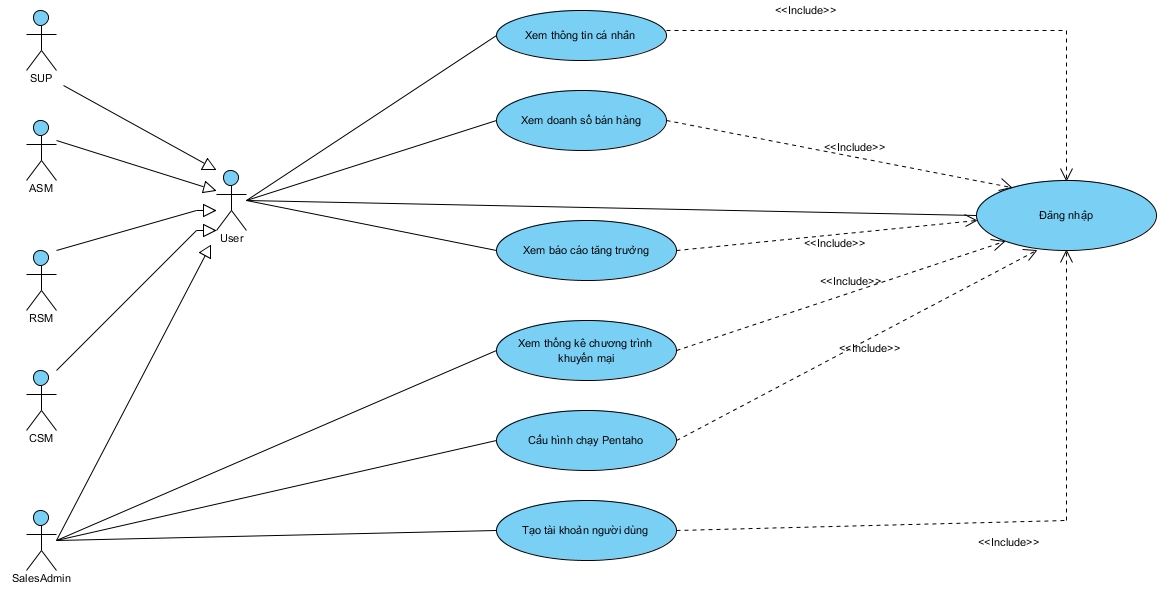
Tiến trình này cần phải được lên lịch trước và nên chạy vào những thời gian không có nhiều người dùng cuối khai thác data warehouse. Thời gian để chạy tiến trình này thường là từ buổi tối cho đến trước giờ làm việc sáng hôm sau. Trong thời gian này số người khai thác data warehouse rất ít, vì vậy nếu người dùng cuối thấy việc khai thác data warehouse là thực sự cần thiết thì thời gian đáp ứng chương trình cũng có thể chấp nhận được. Ngoài ra trong thời gian này thì tài nguyên của hệ thống cũng có rất nhiều, điều này cũng góp phần làm giảm thời gian tải dữ liệu vào data warehouse.

# **Chương 4: Phân tích thiết kế hệ thống**

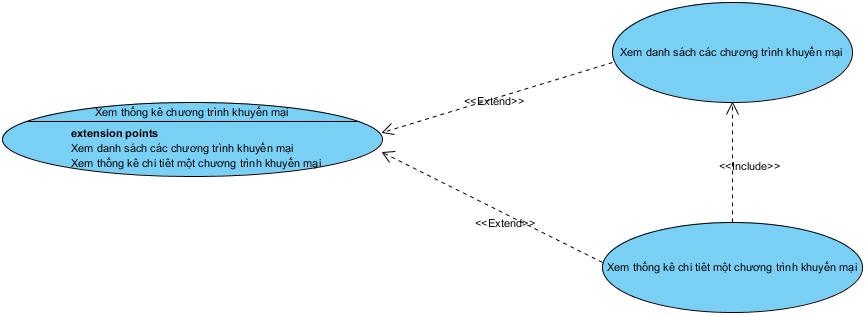
* 1. Phân tích hệ thống

Đồ án sử dụng phương pháp phân tích hướng đối tượng, sử dụng các cú pháp của UML 2.0 và công cụ Visual Paradigm để mô tả hệ thống.

* + 1. Biểu đồ Use Case



Hình 4. 1 Biểu đồ Use case tổng quan



Hình 4. 2 Use case phân rã xem thống kê chương trìn khuyến mại

# **Chương 5: Cài đặt và kiểm thử hệ thống**