**OBJECT – DATABASE SYSTEMS**

**CSDL HƯỚNG ĐỐI TƯỢNG**

**CSDL quan hệ cung cấp một tập nhỏ và cố định các kiểu dữ liệu (như là integers, dates, times…) đã được chứng minh là thích hợp với các ứng dụng truyền thống như xử lý dữ liệu hành chính. Tuy nhiên, trong nhiều ứng dụng, các kiểu dữ liệu phức tạp cũng được sử dụng. Thông thường, những loại dữ liệu phức tạp này được lưu trữ trong hệ thống tập tin OS hoặc được tạo thành dữ liệu có cấu trúc, hơn là lưu trng DBMS. Các loại lĩnh vực có sử dụng kiểu dữ liệu phức tạp bao gồm các mô hình và hệ thống máy tính hỗ trợ như CAD/ CAM (thiết kế với sự hỗ trợ của máy tính/sản xuất với sự hỗ trợ của máy tính), kho đa phương tiện (multimedia) và quản lý tài liệu.**

**Khi số lượng dữ liệu gia tăng, nhiều tính năng được cung cấp bởi DBMS – ví dụ như ứng dụng giảm thời gian phát triển,đồng thời kiểm soát và phục hồi, hỗ trợ lập chỉ mục, và khả năng truy vấn - ngày càng trở nên hấp dẫn và cuối cùng, cần thiết. Để hỗ trợ các ứng dụng như vậy, DBMS phải cung cấp các kiểu dữ liệu phức tạp. Khái niệm hướng đối tượng đã tác động mạnh mẽ đến nỗ lực làm tăng sự hỗ trợ của CSDL cho các dữ liệu phức tạp và đã dẫn đến việc phát triển hệ thống CDSL đối tượng, là chủ đề mà chúng ta thảo luận ở chương này.**

**Hệ thống CSDL đối tượng phát triển theo hai hướng riêng biệt:**

* **Hệ thống CSDL hướng đối tượng: hệ thống CSDL HĐT được đề xuất như là một thay thế cho hệ thống quan hệ và được nhắm vào các lĩnh vực ứng dụng có các đối tượng phức tạp giữ vai trò trung tâm. Cách tiếp cận bị ảnh hưởng nhiều bởi các ngôn ngữ lập trình hướng đối tượng và có thể được hiểu như là một sự cố gắng thêm chức năng DBMS vào môi trường ngôn ngữ lập trình.**
* **Hệ thống CSDL quan hệ đối tượng: hệ thống CSDL quan hệ đối tượng có thể được nghĩ như là một cố gắng để mở rộng CSDL quan hệ với các chức năng cần thiết để hỗ trợ một lớp rộng hơn các ứng dụng, và bằng nhiều cách, tạo một cầu nối giữa các mô hình quan hệ và hướng đối tượng.**

**Ta sẽ sử dụng tên viết tắc cho các hệ thống quản lý: CSDL quan hệ - RDBMS, CSDL hướng đối tượng – OODBMS, CSDL quan hệ đối tượng – ORDBMS. Trong chương này, chúng ta sẽ tập trung vào ORDBMS và nhấn mạnh làm thế nào chúng có thể được xem là sự phát triển của RDBMSs, chứ không phải là một mô hình khác nào đó.**

**Chuẩn SQL:1999 được dựa trên mô hình ORDBMS, chứ không phải mô hình OODBMS. Chuẩn này bao gồm sự hỗ trợ cho nhiều loại dữ liệu phức tạp mà chúng tôi đề cập đến trong chương này. Chúng tôi đã tập trung vào việc phá triển các khái niệm cơ bản, chứ không phải trình bày về chuẩn SQL:1999; một số tính năng mà chúng tôi thảo luận không nằm trong chuẩn SQL:1999. Chúng tôi đã cố gắng để các kí hiệu phù hợp với chuẩn SQL:1999, mặc dù đôi lúc chúng tôi có chút khác biệt cho rõ ràng hơn. Quan trọng là chúng ta phải nhận ra rằng các khái niệm chính được thảo luận ở đây chung cho cả hai mô hình ORDBMS và OODBMS, và chúng tôi sẽ trình bày làm thế nào chúng hỗ trợ cho chuẩn ODL/OQL (Object Definition Language / Object Query Language) được đề nghị cho OODBMS trong mục 25.8.**

**Các nhà cung cấp RDBMS, bao gồm IBM, Informix và Oracle, đang thêm chức năng ORDBMS (để tăng phạm vi) trong sản phẩm của họ, và điều quan trọng là nhận ra làm thế nào các kiến thức hiện có về thiết kế và sự thực thi của CSDL quan hệ có thể được nâng cấp để xử lý các phần mở rộng của ORDBMS. Cũng quan trọng để hiểu rằng những thách thức và cơ hội mà những phần mở rộng đó đưa ra cho những người sửa dụng, người thiết kế và người thực hiện.**

**Trong chương này, mục 25.1 đến 25.5 thúc đẩy và giới thiệu các khái niệm hướng đối tượng. Những khái niệm được thảo luận trong những mục đó đều thuộc về cả hai hệ thống OODBMS và ORDBMS, mặc dù cú pháp của chúng tôi thì tương tự với chuẩn SQL:1999. Chúng tôi sẽ bắt đầu bằng cách đưa ra một ví dụ trong mục 25.1 minh họa cho việc tại sao phần mở rộng cho mô hình quan hệ lại cần thiết để thích ứng với các lĩnh vực ứng dụng mới. Ví dụ này sẽ được sử dụng xuyên suốt chương này. Chúng tôi sẽ thảo luận làm thế nào những kiểu dữ liệu trừu tượng có thể được định nghĩa và xử lý trong mục 25.2 và làm thế nào các kiểu có thể được kết hợp vào các kiểu có cấu trúc trong mục 25.3. Sau đó chúng tôi sẽ xem xét các đối tượng và đối tượng xác định trong mục 25.4 và sự kế thừa và kiểu phân cấp trong mục 25.5.**

**Chúng tôi sẽ xem làm thế nào để tận dụng lợi thế của khái niệm hướng đối tượng mới để tạo bản thiết kế cho CSDL ORDBMS trong 25.6. Trong 25.7, chúng tôi sẽ thảo luận những thách thức trong thực thi trong CSDL quan hệ đối tượng. Thảo luận về ODL và OQL, chuẩn cho OODBMSs trong mục 25.8, và sau đó đưa ra một sự so sánh ngắn gọn về ORDBMS và OODBMS trong 25.9.**

**25.1. VÍ DỤ:**

**Là một ví dụ cụ thể về sự cần thiết cho các hệ thống hướng đối tượng, chúng tôi tập trung vào các vấn đề xử lý dữ liệu trong kinh doanh mới mà chúng tôi cho rằng nó khó hơn và thú vị hơn so với những sổ sách kế toán trong tập kỷ trước. Ngày nay, các công ty trong các ngàng công nghiệp như giải trí đang trong việc kinh doanh bán *bits*; những tài sản cơ bản của công ty học không phải là những sản phẩm hữu hình, mà là những sản phẩm phần mềm như là video hay audio.**

**Chúng tôi đã xem qua công ty giải trí Dinky, một tập đoàn lớn ở Hollywood, nơi nắm giữ một bộ sưu tập các nhân vật hoạt hình, đặc biệt là chú giun Herbert thích được vuốt ve và rất được yêu mến trên thế giới. Dinky có một số phim về chú giun Herbert, nhìu trong số đó đã được trình chiếu trong các rạp trên toàn thế giới tại bất kì thời điểm nào. Dinky còn sử dụng những hình ảnh, âm thanh và những cảnh phim của Herbert cho nhiều mục đích: các nhân vật hành động, video game, sản phẩm quảng cáo… CSDL của Dinky được sử dụng để quản lý doanh thu và những hợp đồng về các sản phẩm liên quan đến Herbert, cũng như các dữ liệu về video và audio để tạo nên nhiều phim về Herbert.**

***25.1.1. Các kiểu dữ liệu mới***

**Một vấn đề cơ bản mà những nhà thiết kế dữ liệu của Dinky gặp phải là họ cần sự hỗ trợ cho các loại dữ liệu phong phú hơn là các dữ liệu có sẵn trong CSDL quan hệ:**

* **Kiểu dữ liệu trừu tượng do người dùng tự định nghĩa (ADTs): tài sản của Dinky bao gồm hình ảnh, âm thanh, video phim của Herbert và chúng cần được lưu trữ trong CSDL. Thêm nữa, cần các chức năng cụ thể để xử lý các đối tượng đó. Ví dụ, chúng tôi có thể muốn viết các chức năng tạo ra các phiên bản nén của một bức ảnh hay các bức ảnh có độ phân giải thấp hơn. (xem 25.2)**
* **Kiểu dữ liệu có cấu trúc: trong ứng dụng này, thực sự trong nhiều ứng dụng xử lý dữ liệu kinh doanh truyền thống, chúng tôi cần các kiểu dữ liệu mới được tạo từ các kiểu nguyên tử sử dụng các kĩ thuật để tạo nên các tập hợp, bộ, mảng, chuỗi… (25.3)**
* **Sự kế thừa: khi số lượng dữ liệu gia tăng, điều quan trọng là nhận ra rằng sự tương đồng giữa các kiểu dữ liệu khác nhau và sử dụng các lợi ích từ nó. Ví dụ, các bức ảnh nén và các bức ảnh có độ phân giải thấp, mở một mức độ nào đó, cũng là hình ảnh. Vì thế, cần biết kế thừa các đặc điểm các đối tượng hình ảnh trong khi định nghĩa (và sau đó là xử lý) đối tượng hình ảnh nén và đối tượng hình ảnh phân giải thấp.**

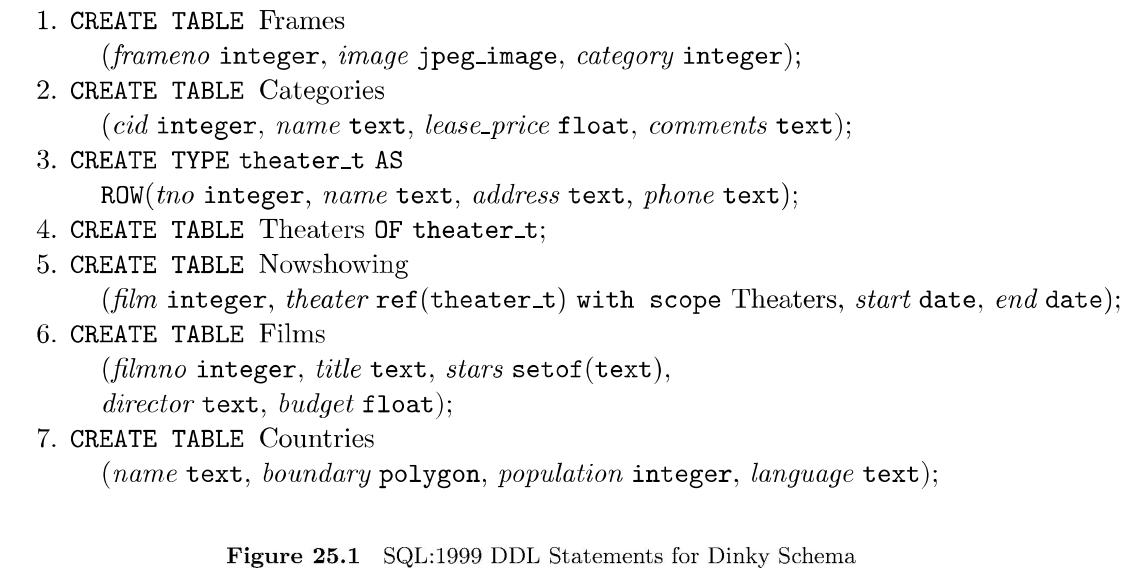
**Làm thế nào để chúng ta giải quyết các vấn đề này trong RDBMS? Chúng tôi có thể lưu trữ hình ảnh, video… dưới dạng BLOBs trong hệ thống quan hệ hiện tại. Binary large object (BLOB) – đối tượng nhị phân lớn chỉ là một chuỗi các byte, và sự hỗ trợ của DBMS bao gồm sự lưu trữ và lấy thông tin ra theo cách mà người sử dụng không phải lo lắng vè kích thước của BLOB; BLOB có thể trải trên nhìu trang, không giống với các thuộc tính truyền thống. Tất cả các xử lý sau đó của BLOB được thực hiện bởi chương trình ứng dụng của người dùng, trong ngôn ngữ máy chủ nơi mã SQL được lưu vào. Giải pháp này không bị ảnh hưởng bởi vì chúng tôi được phép lấy toàn bộ BLOBs trong bộ sưu tập cho dù hầu hết chúng có thể được thâm nhập bằng cách sử dụng chức năng tự định nghĩa của người dùng (bao gồm DBMS). Nó không phù hợp với quan điểm nhất quán dữ liệu, bởi vì ngữ nghĩa của dữ liệu bị phụ thuộc nhiều vào mã ứng dụng ngôn ngữ máy chủ và không thể được thực thi bởi DBMS.**

**Đối tượng lớn trong SQL: chuẩn SQL:1999 gồm kiểu dữ liệu mới gọi là LARGE OBJECT hay LOB với hai biến thể gọi là BLOB (binary large object) và CLOB (character large object). Nó chuẩn hóa sự hỗ trợ đối tượng lớn được tìm thấy trong nhiều CSDL quan hệ hiện nay. LOBs có thể không bao gồm khóa chính, các mệnh đề GROUP BY hay ORDER BY. Chúng có thể được so sánh sử dụng đẳng thức, bất đẳng thức và các phép tính chuỗi. LOB có một locator về cơ bản đó là một id duy nhất và cho phép LOBS được xử lý mà không cần bản mở rộng.**

**LOBS thông thường được lưu trữ riêng rẽ trong các bộ dữ liệu, nơi mà các trường của nó xuất hiện. IBM DB2, Informix, Microsoft SQL Server, Oracle 8 và Sybase ASE đều hỗ trợ LOBs.**

**Với kiểu có cấu trúc và kế thừa, đơn giản là không có hỗ trợ nào trong mô hình quan hệ. Chúng tôi buộc phải sắp xếp dữ liệu với cấu trúc phức tạp trong một bộ các bảng phẳng. (Chúng tôi đã thấy những ví dụ về các ánh xạ khi chúng tôi thảo luận về các bản chuyển từ mô hình ER với sự kế thừa đến qua mô hình quan hệ trong Chapter 2)**

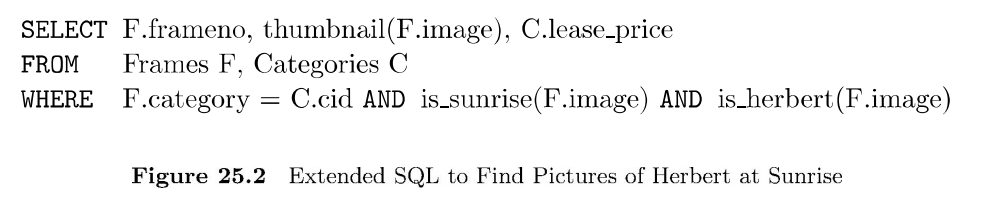
**Ứng dụng này rõ ràng yêu cầu các chức năng không có trong mô hình dữ liệu. Như một ví dụ của những chức năng này, mục 25.1 trình bày chuẩn SQL:1999 DDL (data definition language) cho lược đồ ORDNMS của Dinky, lược đồ nãy sẽ được sử dụng trong ví dụ sau. Mặc dù DDL tương tự với hệ thống quan hệ truyền thống, nhưng nó có vẫn có những sự khác biệc quan trọng làm nổi bật khả năng mô hình dữ liệu của ORDBMS. Nhìn nhanh qua DDL như vậy là đủ; chúng tôi sẽ nghiên cứu chúng chi tiết trong phần tiếp theo, sau khi trình bày một số khái niệm cơ bản mà phần mềm đề đơn giản của chúng tôi cần trong CSDL tiếp theo.**

****

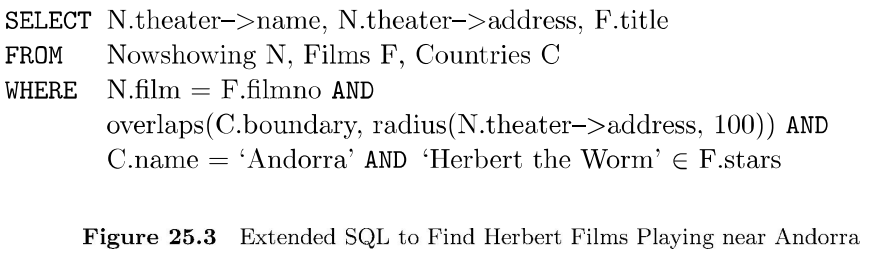
***25.1.2. Xử lý những loại dữ liệu mới***

**Chúng tôi sẽ mô tả loại dữ liệu mới cần được lưu trữ trong csdl của Dinky. Chúng tôi không nói bất kỳ điều gì về các sử dụng những kiểu mới này trong câu truy vấn, vì vậy hãy nghiên cứu hai câu truy vấn mà CSDL của Dinky cần phải hỗ trợ. Cú pháp của các câu truy vấn không quan trọng, cú pháp này đủ để hiểu chúng muốn thể hiện cái gì. Chúng tôi sẽ trở lại những đặc điểm của cú pháp câu truy vấn khi chúng tôi đến phần này.**

**Thử thách đầu tiên của chúng tôi đến từ công ty ngũ cốc ăn sáng Clog. Clog sản xuất một loại ngũ cốc gọi là Delirios, và họ muốn thuê hình ảnh của Herbert trong buổi bình minh, để kết hợp với mẫu thiết kế hộp Delirios. Một câu truy vấn để trình bày một bộ sưu tập các hình ảnh và giá cho thuê của chúng có thể được thể hiện trong SQL – xem Figure 25.2.Dinky có một số phương pháp viết bằng ngôn ngữ mệnh lệnh như Java và được bảo đảm trong hệ thống CSDL. Những phương pháp này có thể được sử dụng trong câu truy vấn theo cách tương tự như các phương pháp đã được xây dựng, như =, +, -, <, >, được sử dụng trong ngôn ngữ quan hệ giống như SQL. Phương pháp** *thumbnail* **trong mệnh đề *Select* tạo ra một phiên bản nhỏ của hình ảnh ban đầu. Phương pháp** *is\_sunrise* **là một hàm kiểu Boolean dùng để phân tích một bức ảnh và trả về** *true* **nếu bức ảnh chứa cảnh bình mình; phương pháp** *is\_herbert* **trả về** *true* **nếu bức ảnh chứa hình ảnh của Herbert. Câu truy vấn tạo ra mã của ảnh, hình ảnh thu nhỏ và giá cho tất cả các ảnh chứa Herbert và bình minh.**

****

**Thử thách thứ hai đến từ ban quản trị của Dinky. Họ biết rằng Delirios cực kì phổ biết ở những nước nhỏ của Andorra, vì thế họ muốn đảm bảo rằng số phim về Herbert sẽ được chiếu ở các rạp gần Andorra khi loại ngũ cốc này đã có mặt trên các kệ của cửa hàng. Để kiểm tra tình trạng của vấn đề, các nhà điều hành muốn tìm tên của tất cả các rạp chiếu phim về Herbert trong bán kính 100km của Andorra. Hình 25.3 thể hiện câu truy vấn theo cú pháp SQL.**

****

**Thuộc tính** *theater* **của bảng Nowshowing là một tham chiếu đến một đối tượng trong một bảng khác có thuộc tính** *name***,** *address* **và** *location***. Đối tượng tham chiếu này cho phép sử dụng ký hiệu** *N.theater -> name* **và** *N.theater -> address***, chúng đại điện cho các thuộc tính của đối tượng *theater\_t* tham chiếu từ bảng Nowshowing. Thuộc tính** *stars* **của bảng** *films* **là một tập các tên của các ngôi sao của phim. Phương pháp radius trả về một vòng tròn mà tâm của nó là đối số đầu tiên của và phạm vi là đối số thứ hai. PP *overlaps* kiểm tra sự chồng chéo không gian. Vì thế, bảng Nowshowing và Films được kết với nhau bằng mệnh đề bằng, trong khi bảng Nowshowing và Countries được kết bởi mệnh đề chồng chéo không gian. Chọn ra “Andorra” và phim chứa “Herbert the Worm” sẽ hoàn thành câu truy vấn.**

**Hai câu truy vấn quan hệ đối tượng này tương tự như câu truy vấn SQL-92 nhưng có một số đặc điểm đáng chú ý:**

* **Phương pháp người dùng định nghĩa: kiểu trừu trượng người dùng định nghĩa được xử lý thông qua phương pháp của chúng, ví dụ như** *is\_herbert* **(section 25.2).**
* **Các toán tử cho kiểu có cấu trúc: cùng với kiểu có cấu trúc có sẵn trong mô hình dữ liệu, ORDBMS cung cấp các phương pháp tự nhiên cho các loại dữ liệu này. Ví dụ, kiểu setof có phương pháp tạo các toán tử (section 25.3.1)**
* **Toán tử cho loại tham chiếu: kiểu tham chiếu sử dụng kí hiệu mũi tên -> (25.3.2)**

**Tóm tắt lại các điểm đáng chú ý bằng ví dụ của chúng tôi, hệ thống quan hệ truyền thống có những hạn chế trong những kiểu dữ liệu có sẵn. dữ liệu được lưu trữ trong các bảng, và kiểu của giá trị mỗi trường bị giới hạn bởi những kiểu đơn giản (integer, string) có kích thước nhỏ và cố định. Các kiểu có giới hạn này có thể được mở rộng theo ba cách: kiểu dữ liệu trừu tượng người dùng định nghĩa, kiểu có cấu trúc và kiểu tham chiếu. Nhìn chung, chúng tôi xem những kiểu mới đó là những kiểu phức tạp. Trong những phần còn lại của chương này, chúng tôi sẽ xem làm thế nào DBMS có thể được mở rộng để hỗ trợ cho việc định nghĩa các kiểu dữ liệu phức tạp mới và xử lý các đối tượng của những kiểu dữ liệu mới đó.**

**25.2. KIỂU DỮ LIỆU TRỪU TƯỢNG NGƯỜI DÙNG ĐỊNH NGHĨA**

**Xem xét bảng Frames trong hình 25.1. Cột** *image* **với kiểu dữ liệu *jpeg\_image* dùng để lưu trữ hình ảnh nén đại diện cho khung nhìn đơn giản của một bộ phim. Kiểu jpeg\_image không phải là kiểu dữ liệu có sẵn trong DBMS mà được định nghĩa bởi người dùng cho ứng dụng lưu trữ dữ liệu hình ảnh nén sử dụng chuẩn JPEG của công ty Dinky. Một ví dụ khác, bảng Countries được định nghĩa trong dòng 7 của trong hình 25.1 có cột** *boundary* **với kiểu *polygon,* dùng để chứa các hình ảnh phát thảo các nước trên bản đồ thế giới. DBMS cho phép người dùng lưu và lấy ra các đối tượng của kiểu *jpeg\_image*, giống như đối tượng của bất kì kiểu dữ liệu khác như *integer*. Các kiểu dữ liệu nguyên tử mới thường cần có toán tử riêng biệt được định nghĩa bởi người dùng tạo nên chúng. Ví dụ, một người có thể định nghĩa toán tử cho kiểu dữ liệu hình ảnh như *compress, rotate, shrink*, và *crop*. Sự kết hợp của các kiểu dữ liệu nguyên tử và các phương pháp liên quan của nó được gọi là kiểu dữ liệu trừu tượng – ADT (abstract data type). Các kiểu dữ liệu truyền thống của SQL cũng có các ADTs như integer (với các phương pháp toán học liên quan), hay strings (với các phương pháp so sánh bằng, LIKE). Hệ thống quan hệ đối tượng bao gồm các ADT này và cũng cho phép người dùng định nghĩa các ADT của họ.**

**Nhãn ‘abstract’ được áp dụng cho cá kiểu dữ liệu này bởi vì hệ CSDL không cần biết làm thế nào dữ liệu của ADT được lưu trữ hay các phương pháp của ADT làm việc như thế nào. Nó đơn thuần chỉ cần biết phương pháp đã có sẵn cái gì và kiểu dữ liệu vào và ra cho phương pháp đó. Phần ẩn của bản chất ADT được gọi là sự chứa đựng // gói gọn // tóm lược(encapsulation). Chú ý rằng ngay cả trong hệ thống quan hệ, kiểu nguyên tố như integer có các phương pháp liên quan cho ADT cũng được chứa trong ADT. Trong trường hợp của integers, phương pháp chuẩn cho ADT là các toán tử và các bộ so sánh toán học thông thường. Để tính giá trị cho toán tử cộng trong integer, hệ CSDL không cần phải hiểu luật của phép cộng – nó chỉ cần biết làm thế nào để gọi đoạn mã của toán tử cộng và kiểu dữ liệu nào sẽ chứa kết quả trả về.**

**Trong hệ thống quan hệ đối tượng, sự đơn giản hóa để gói gọn là rất quan trọng bởi vì nó giấu đi bất kỳ sự khác biệt riêng rẽ giữa các kiểu dữ liệu và cho phép ORDBMS được thực hiện mà không cần biết trước kiểu và phương pháp mà người dùng có thể muốn thêm vào. Ví dụ, cộng integer và che hình ảnh có thể được xử lý giống nhau bởi hệ thống, với sự phân biệt quan trọng duy nhất, hai đoạn mã khác nhau sẽ được gọi cho hai phép tính và các kiểu đối tượng khác nhau sẽ được trả về từ đoạn code đó.**

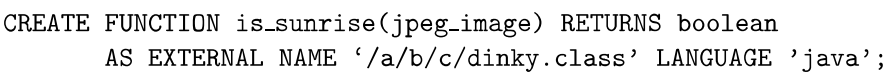
**Gói ORDBMS mở rộng: sự phát triển của tập các kiểu và pp do người dùng định nghĩa cho các ứng dụng đặc biệt - ví dụ như quản lý hình ảnh – có thể bao gồm một số lượng công việc đáng kể và chuyên môn của lĩnh vực đặc trưng. Kết quả là hầu hết các nhà cung cấp ORDBM thường hợp tác với bên thứ 3 để bán các tập hợp các gói ADT cho các lĩnh vực riêng biệt. Informix gọi những phần mở rộng này là** *DataBlades***, Oracle gọi chúng là** *Data Cartridges***, IBM gọi là** *DB2 Extengers***… Những gói này bao gồm đoạn mã của pp ADT, bản DDL để tự động tải ADT vào hệ thống, và trong một số trường hợp là pp truy cập chuyên biệt cho kiểu dữ liệu. Gói ADT mở rộng tương tự với thư việc lớp có sẵn trong ngôn ngữ lập trình hướng đối tượng: chúng cung cấp một tập hợp các đối tượng cùng nhau giải quyết một công việc chung.**

***25.2.1. Định nghĩa phương pháp của ADT***

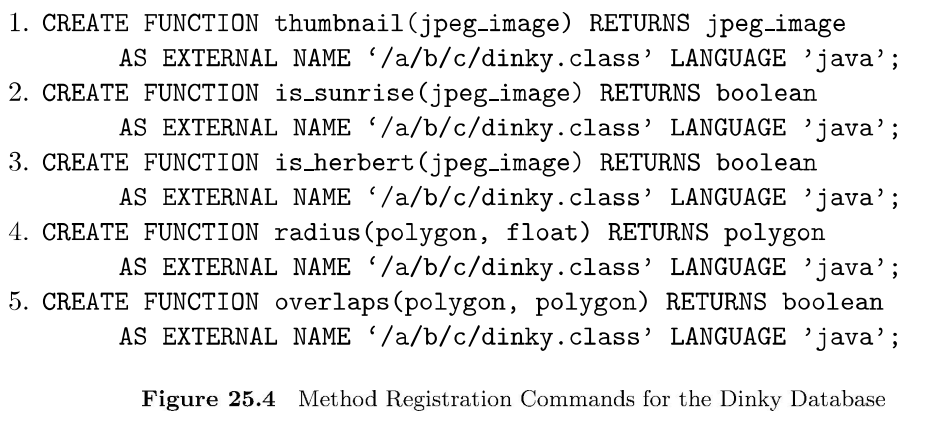
**Tại điểm tối tiểu, với mỗi kiểu nguyên tử mới, người dùng cần phải định nghĩa pp cho phép DBMS có thể đọc và lấy đối tượng của kiểu đó và tính toán số lượng cần lưu trữ để chứa đối tượng. Người dùng tạo ra kiểu nguyên tử mới phải đăng kí các pp sau với DBMS:**

* **Size: trả về số byte lưu trữ yêu cầu cho đối tượng của kiểu hay giá trị biến thiên, nếu nó thay đổi theo độ lớn.**
* **Import: tạo một đối tượng của kiểu này từ bản đầu vào (vd: INSERT statements).**
* **Export: nối đối tượng của kiểu này với một lớp thích hợp để in hay để sử dụng cho chương trình ứng dụng (vd: chuỗi ASCII hay tập tin xử lý).**

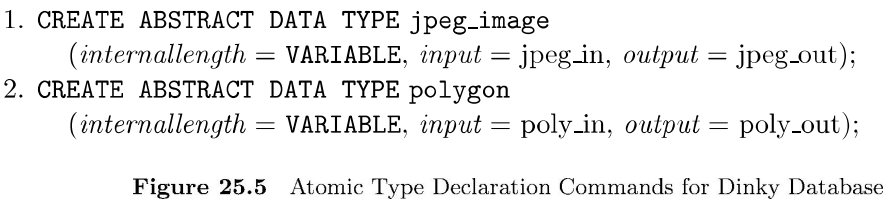
**Để đăng kí một pp mới cho kiểu nguyên tố, người dùng phải viết đoạn mã cho pp và sau đó khai báo với hệ CSDL về pp này. Đoạn mã được viết dựa trên ngôn ngữ được hỗ trợ bởi DBMS, và hệ thống có thể xử lý được vấn đề. Ví dụ, ORDBMS có thể sử dụng mã Java trong hệ thống điều khiển Linux. Trong trường hợp này, mã pp phải được viết bằng Java và biên dịch thành một file mã byte Java được lưu trữ trong ORDBMS,vì thế nó có thể nhận ra pp mới:**

****

**Mệnh đề này định nghĩa các khía cạnh nổi bật của pp: kiểu của ADT liên quan, kiểu trả về, và vị trí của đoạn mã. Khi pp đã được đăng kí, DBMS sử dụng máy ảo Java để thực thi đoạn mã. Hình 25.4 trình bày một số pp đã đăng kí cho CSDL của Dinky:**

****

**Các mệnh đề định nghĩa cho các kiểu dữ liệu người dùng định nghĩa trong lược đồ Dinky được trình bày trong hình 25.5.**

****

***25.3. Kiểu có cấu trúc***

**Kiểu nguyên tử và kiểu do người dùng định nghĩa có thể được kết hợp để mô tả một cấu trúc phức tạp hơn sử dụng type constructors (kĩ thuật phân loại). Ví dụ, dòng 6 của hình 25.1 định nghĩa cột** *stars* **của kiểu *setof(text)*; mỗi entry trong cột là một tập hợp các dòng văn bản, đại diện cho các ngôi sao của phim. Cú pháp của *setof* là một ví dụ của type constructors. Một type constructors thông thường khác bao gồm:**

* *ROW (n1 t1, … , nntn ):* **kiểu này đại diện cho hàng, hay bộ của n trường với các trường** *n1, …, nn* **của các kiểu** *t1, …, tn* **tương ứng.**
* ***listof (base):* một kiểu đại diện cho một chuỗi các kiểu *base*.**
* ***ARRAY (base)*: một kiểu đại diện cho một mảng các kiểu *base*.**
* ***setof (base)*: một kiểu đại diện cho một tập hợp các kiểu *base*. Các tập hợp không thể chứa các phần tử giống nhau.**
* ***bagof (base)*: một kiểu đại diện cho** *bag* **hay** *multiset* **của kiểu *base*.**

**Kiểu dữ liệu có cấu trúc trong SQL: kiểu theater\_t trong hình 25.1 minh họa cho kiểu dữ liệu ROW trong SQL:1999; một giá trị của kiểu ROW có thể xuất hiện trong một trường của một bộ. trong SQL:1999, kiểu ROW có một vai trò đặc biệt bởi vì mỗi bảng là một bộ các dòng – mỗi bảng là một tập hợp các dòng hay các đa tập hợp của các dòng. SQL\_1999 cũng bao gồm kiểu dữ liệu gọi là ARRAY cho phép giá trị của một trường vào một mảng. Kiểu ROW và ARRAY có thể xen kẽ và lồng vào nhau để xây dựng cấu trúc của đối tượng. Các kiểu *listof, bagof* và *setof* không thuộc SQL:1999. IBM DB2, Informix UDS và Oracle 8 đều hỗ trợ cấu trúc ROW.**

**Để hiểu đầy đủ sức mạnh của pp xây dựng kiểu, hãy quan sát chúng có thể được kết hợp; ví dụ, ARRAY ( ROW (age: integer, sal: integer)). Kiểu được định nghĩa sử dụng pp xây dựng kiểu được gọi là kiểu có cấu trúc. Chúng sử dụng listof, ARRAY, bagof hay setof như các kiểu ngoài cùng đôi khi được gọi là collection types hay bulk data types.**

**Sự giới thiệu về dữ liệu có cấu trúc thay đổi đặc điểm cơ bản của CSDL quan hệ, đó là tất cả các trường đều chứa các giá trị nguyên tử. Một quan hệ chứa đối tượng có cấu trúc không phải là mẫu bình thường đầu tiên. Chúng tôi sẽ thảo luận điểm này sau trong mục 25.6.**

***25.3.1. Xử lý kiểu có cấu trúc của dữ liệu***

**DBMS cung cấp các pp đã được xây dựng cho các kiểu được hỗ trợ thông qua pp xây dựng. Các pp này tương tự như các phép tính đã có sẵn như cộng, nhân cho kiểu nguyên tử như integer. Trong phần này chúng tôi sẽ trình bày cách xây dựng các kiểu khác nhau và minh họa làm thế nào các câu truy vấn SQL có thể tạo và xử lý các giá trị của kiểu có cấu trúc.**

**Toán tử gắn liền cho kiểu dữ liệu có cấu trúc**

**Bây giờ chúng tôi sẽ xem các toán tử đi kèm cho mỗi kiểu có cấu trúc mà chúng tôi đã trình bày trong mục 25.3.**

**Rows: cho 1 đối tượng I mà kiểu của nó là ROW (***n1 t1, …, nntn***), pp khai thác các trường cho phép chúng ta truy cập riêng rẽ một trường** *nk***bằng cách sử dụng ký hiệu dấu chấm** *i.nk***. Nếu các cấu trúc dòng lồng vào các định nghĩ kiểu, thì dấu chấm có thể sẽ được lồng vào nhau để truy cập các trường của của các dòng lồng nhau; ví dụ như** *i.nk.ml***. Nếu chúng tôi có một bộ các hàng, ký hiệu dấu chấm cho chúng tôi kế quả là một bộ sưu tập. Ví dụ, nếu i là một danh sách các dòng,** *i.nk* **cho chúng tôi một danh sách các đối tượng có kiểu** *tn* **; nếu i là một tập các dòng,** *i.nk* **cho chúng tôi một tập các đối tượng của kiểu** *tn* **.**

**Các ký hiệu dấu chấm lồng nhau này thường được gọi là path expression (biểu thức dẫn) bởi vì nó được mô tả là một đường dẫn đến cấu trúc lồng nhau.**

**Sets and multisets: đối tượng cố định có thể được xem như là sử dụng pp truyền thống . Một đối tượng của kiểu *setof (foo)* có thể được xem như là một đối tượng của kiểu foo sử dụng pp , như minh họa trong hình 25.3 chứa nét tương đồng của “***Herbert the Worm***”** *F.stars***. Hai đối tượng (có các phần tử của kiểu dữ liệu giống nhau) có thể được kết hợp để tạo nên một đối tượng mới sử dụng toán tử và .**

**Mỗi pp của sets bộ có thể được định nghĩa cho multisets, lấy số lượng các bản sao của các phần tử vào trong tài khoản. Toán tử đơn giản chỉ thêm một số bản sao của các phần tử, toán tử đếm số các phần tử có trong cả hai multisets đầu vào, và toán tử trừ số phần tử xuất hiện trong multiset thứ 2 trong các bộ của multiset thứ nhất. Ví dụ:**

****

****

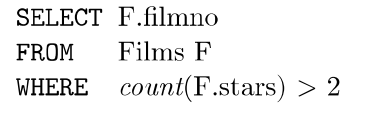
**Lists: các phương thức của list truyền thống gồm head, trả về phần tử đầu tiên; tail, trả về list bằng cách bỏ đi phần tử đầu tiên; prepend, lấy một phần tử và thêm nó vào vị trí của phần tử đầu tiên của list; append, ghép một list vào list khác.**

**Array: kiểu mảng hỗ trợ đánh chỉ mục cho mảng cho phép người sử dụng truy cập các phần tử trong mảng at a particular offset. Cú pháp ‘square bracket’ thường được sử dụng; ví dụ, foo\_Array[5].**

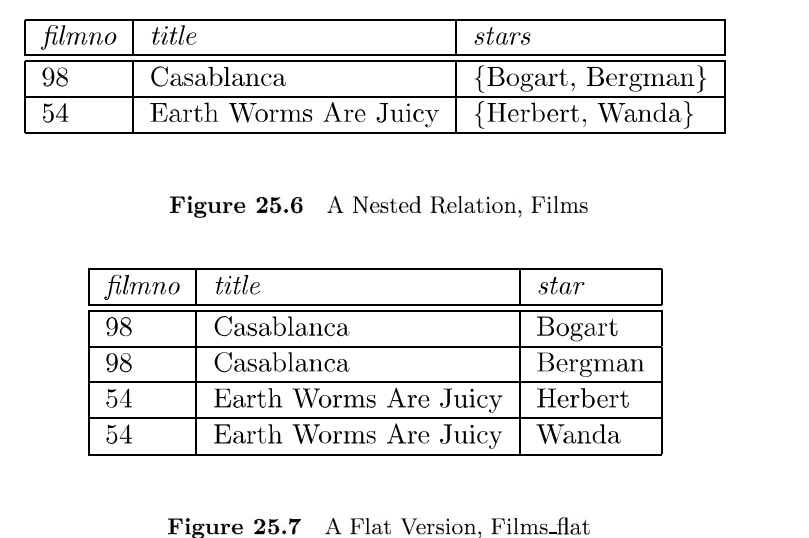
**Các kiểu khác: các toán tử đã nêu ở trên chỉ là đơn giản. Chúng tôi cũng có những toán tử như** *count, sum, avg, max, min***, có thể được áp dụng vào bất kì đối tượng nào của bộ sưu tập dữ liệu. Các toán tử cho kiểu chuyển đổi cũng phổ biến. Ví dụ, chúng tôi có thể cung cấp toán tử để chuyển một đối tượng multiset sang đối tượng có kiểu set bằng bỏ các phần giống nhau.**

**Ví dụ của các câu truy vấn chứa các bộ lồng nhau**

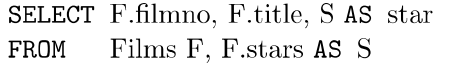
**Bây giờ chúng tôi sẽ trình bày ví dụ để minh họa làm thế nào các quan hệ chứa các bộ trùng nhau có thể được truy vấn, sử dụng cú pháp SQL. Hãy xem quan hệ Films. Mỗi bộ mô tả một phim, mã xác định duy nhất là filmno, và chứa một tập của các ngôi sao trong phim như là giá trị của trường. Ví dụ đầu tiên của chúng tôi sẽ minh họa làm thế nào chúng tôi có thể ứng dụng một toán tử tính toán vào các bộ lồng nhau. Nó nhận biết các phim với hơn hai sao bằng cách đếm số sao; toán tử** *count* **được ứng dụng trên mỗi bộ của Films.**

****

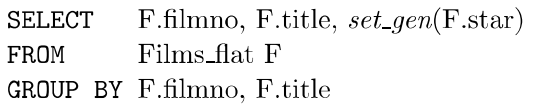
**Câu truy vấn thứ hai của chúng tôi minh họa cho biểu thức gọi là unnesting. Hãy xem ví dụ của Films trong hình 25.6; chúng tôi đã bỏ qua trường director và budget (trong hình 25.1) để đơn giản hơn. Một bản phẳng của các thông tin giống nhau được thể hiện trong hình 25.7; với mỗi phim và sao của phim, chúng tôi có bộ Films\_flat.**

****

**Câu truy vấn dưới đây phát sinh từ ví dụ của Films\_flat từ Films:**

****

**Biến F đại diện cho các bộ trong bảng Films, và với mỗi giá trị của F, biến S đại diện cho tập hợp các giá trị trong trường** *stars* **của F. Ngược lại, chúng tôi muốn phát sinh một bản khác của Films từ Films\_flat. Chúng tôi có thể phát sinh bảng Films bằng cách sử dụng một bản tổng quát hóa của cấu trúc GROUP BY trong SQL, như câu truy vấn minh họa sau:**



***Objects và oid***s: **trong chuẩn SQL:1999, mỗi bộ trong một bảng có thể được cho một oid bằng cách định nghĩa bảng trong kiểu có cấu trúc, như sự định nghĩa của bảng Theaters trong dòng 4 trong hình 25.1. So sánh với sự định nghĩa của bảng Countries trong dòng 7; các bộ Countries không có oids. Trong chuẩn SQL:1999 cũng gán oid cho các đối tượng lớn: đây là mã xác định của đối tượng (định danh).**

**Có một kiểu đặc biệt gọi là REF mà giá trị của nó được xác định duy nhất. SQL:1999 yêu cầu kiểu REF phải được gắn với kiểu có cấu trúc riêng biệt và bảng mà nó tham chiếu phải được biết vào thời điểm biên dịch, phạm vi của mỗi tham chiếu phải là một bảng đã được xác định ở thời điểm biên dịch. Ví dụ, dòng 5 trong hình 25.1 định nghĩa cột** *theater* **với kiểu *ref(theater\_t).* Đối tượng trong cột được tham chiếu đến đối tượng của kiểu theater\_t, nói đúng hơn là dòng trong bảng Theater đã được định nghĩa trong dòng 4. IBM DB2, Informix UDS và Oracle 8 đều hỗ trợ kiểu REF.**

**Toán tử set\_get, được dùng với GROUP BY, yêu cầu một số lời giải thích. Mệnh đề GROUP BY chia bảng Films\_flat bằng cách phân loại trên thuộc tính** *filmno***; tất cả các bộ trong cùng một loại có chung mã** *filmno* **(và do đó có cùng** *title***). Hãy xem tập hợp các giá trị rtong cột star của một loại nào đó. Tập hợp này không thể được trả về bằng kết quả của câu truy vấn trong SQL-92, và chúng tôi phải lấy nó bằng cách áp dụng toán tử tập hợp (aggregate) như COUNT. Bây giờ chúng tôi cho các quan hệ chứa các tập hợp là giá trị của các trường, tuy nhiên, chúng tôi muốn trả về tập giá trị star như là giá trị của trường trong một bộ đơn lẻ; bộ trả về cũng chứa gía trị filmno của phân loại tương ứng. Toán tử set\_gen tập hợp các giá trị star trong một loại và tạo ra đối tượng giá trị. Phép tính này gọi là *nesting*. Chúng tôi có thể tưởng tượng các chức năng phát sinh tương tự để tạo multiset, list,… Tuy nhiên, những chức năng phát sinh này không nằm trong SQL:1999.**

**25.4. OBJECTS, OBJECTS IDENTIFY, AND REFERENCE TYPES**

**Trong hệ thống CSDL đối tượng, đối tượng dữ liệu có thể được xem như là *object identifier* (oid), một số giá trị là duy nhất trong CSDL. DBMS chiệu trách nhiệm phát sinh các oid và đảm bảo rằng các oid dùng để nhận dạng các đối tượng duy nhất trong suốt quá trình tồn tại của nó. Trong một số hệ thống, tất cả các bộ được lưu trữ trong các bảng là các đối tượng và tự động được gắn với định danh duy nhất; trong các hệ thống khác, người dùng có thể tự xác định định danh cho các bộ của các bảng. Thỉnh thoảng, cũng có khả năng phát sinh oid cho các cấu trúc lớn hơn (bảng) cũng như các cấu trúc nhỏ(tập các giá trị dữ liệu như bản sao của giá trị integer 5, hay hình ảnh JPEG).**