Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Institut für Informationssysteme, Dr. Markus Tresch
Uwe Röhm

Verteilte Objektverwaltung: Client, Server & Middleware
Aufgabenblatt 5

Data Warehousing / Sichtenwartung

Szenario

Wir möchten Ihnen unsere kleine Fluggesellschaft *DOMinion Airlines* vorstellen, deren Firmendatenbestand in drei verschiedenen Datenbanken gespeichert ist.

Es gibt eine *PersonalDB*, in welcher in der Klasse `Pilots` die Personaldaten aller Piloten verwaltet sind. Neben der Personalnummer `PNr` sind das unter anderem der `Name`, die `Adresse` und die Menge `Licenses` von Flugzeugtypen, für die ein Pilot ausgebildet ist.

Der Maschinenpark von DOMinion Airlines wird mittels der Datenbank *InventoryDB* verwaltet. Uns interessiert vorallem die Klasse `Planes` der vorhandenen Flugzeuge. Jedes Flugzeug hat eine Inventarnummer, eine Modellbezeichnung, einen Hersteller, Anzahl von Sitzen und ist für eine Menge von Flugklassen geeignet (`Usages`).

Zu guter letzt gibt es noch die Datenbank *BusinessDB*, über die der laufende Flugbetrieb abgewickelt wird. Hier findet sich die Klasse `Flights`, in der alle Flugverbindungen enthalten sind, die DOMinion Airlines anbietet. Zu den Flügen ist gespeichert, welche Flughäfen sie verbinden, und jedes Flight-Objekt bietet eine Methode `Classify()`, die die Art eines Flugs (z.B. Mittelstrecke, Transkontinental etc) ergibt.

Zusätzlich unterhält DOMinion Airlines ein Data Warehouse, in dem sie verschiedene Firmenübergreifende Daten zusammenfasst. Wir betrachten hier die Sicht *Connections*, die zu den Flugverbindungen die Menge der Piloten materialisiert, die aufgrund ihrer Lizenzen dort eingesetzt werden können. Diese Sicht wird definiert durch:

```
create view Connections as
      select *, Pilots := ( select p                          // unique !
                            from  Pilots@PersonalDB p,
                                  Planes@InventoryDB a
                            where Classify(f) in Usages(a)
                            and   Model(a) in Licenses(p) )
      from Flights@BusinessDB f
```

Aufgaben

1. Bezüglich welcher Operationen (*Create(obj, ...), Delete(obj), Update(obj, attr, old, new)*) auf den drei zugrundeliegenden Datenbanken ist die Data Warehouse Sicht *Connections* selbstwartbar?

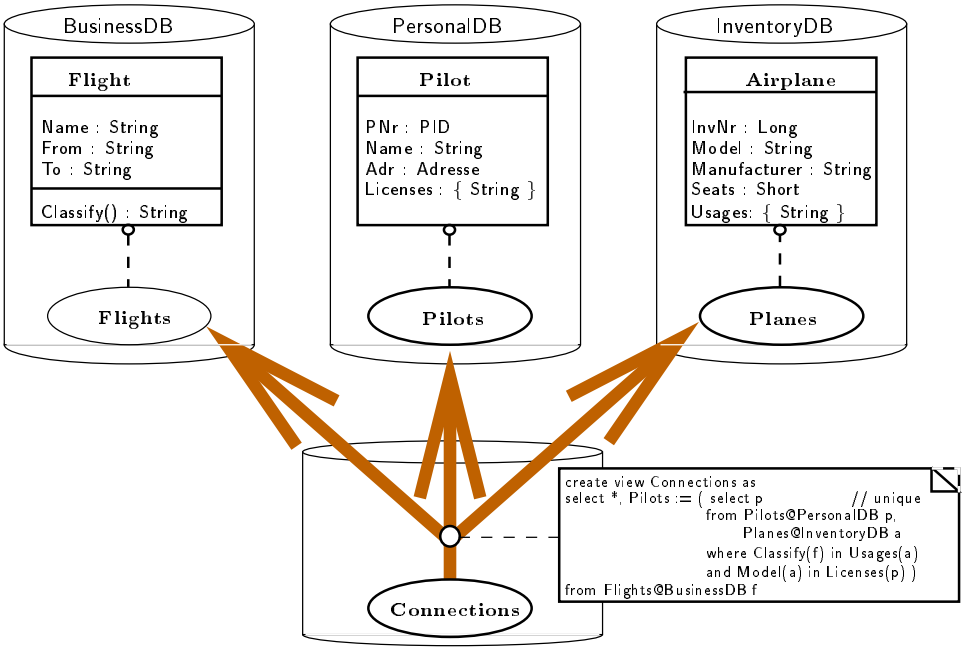

Abbildung 1: Die Struktur des Data Warehouse von DOMinion Airlines

2. Der Integrator des Data Warehouse verwendet Regeln, um auf gemeldete Datenänderungen zu reagieren und die Sicht `Connections` zu aktualisieren. Für das Erzeugen eines neuen Flugs heisst die Regel beispielsweise:

```
on create (oid, Name, From, To) in Flights@BusinessDB do
    create (oid, Name, From, To, Pilots := ( select p
                                             from   Pilots@PersonalDB p,
                                                    Planes@InventoryDB a
                                             where  Classify(oid) in Usages(a)
                                             and    Model(a) in Licenses(p) ))
    in Connections
```

   Formulieren Sie auch für alle anderen Änderungsoperationen die Wartungsregeln des Integrators in einer Syntax Ihrer Wahl, die an das Beispiel angelehnt ist. Welche Informationen müssen die Monitore jeweils dem Integrator mindestens melden?

3. Da das neue Attribut `Pilots` der globalen Sicht `Connections` eine echte Menge ist, also keine Duplikate enthält, ist die Sicht zumindest bezüglich einer Änderungsoperationen in einer Datenbanken nicht selbstwartbar. Dies lässt sich ändern, indem man einen Zähler für jedes Element der Pilotenmenge einführt, der angibt, wie häufig ein Pilot-Objekt sich für ein Connection-Objekt qualifiziert hat. Bezüglich welcher Operationen ist `Connections` nun selbstwartbar, und wie wird dazu der Zähler eingesetzt?

Ausgabe und Vorbesprechung: 14.1.1997
Abgabe und Besprechung: 21.1.1997