AG Heterogene Informationssysteme
Prof . Dr.-Ing. Stefan Deßloch
Fachbereich Informatik
Technische Universität Kaiserslautern

# Recent Developments for Data Models – Solution to Exercise 1

Monday, May 07, 2012 – 15:30 to 17:00 – Room 36-336

## 1) User-defined Structured Types and Typed Tables

In this exercise[1], a database to store publications of a research group is considered. The conceptual schema is depicted in Figure 1 using an UML class diagram.

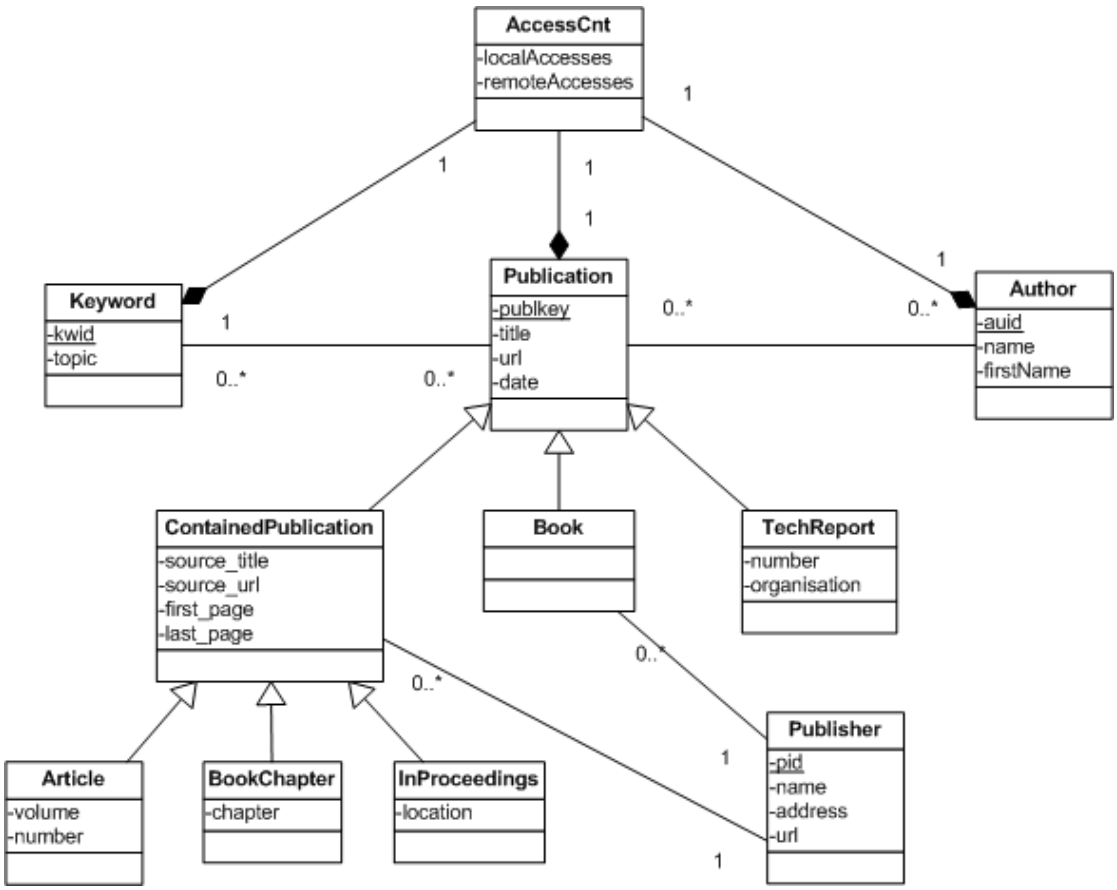


**Figure 1: Conceptual schema of the publication database**

[1] This exercise is based on *Andreas Geppert: Objektrelationale und objektorientierte Datenbankkonzepte und –systeme, dpunkt.verlag, 2002*.

Different types of publications are to be distinguished: Articles, book chapters, and proceeding papers are contained in collections of publications, whereas books and technical reports are not. Each publication is written by one or more authors. Authors may contribute to multiple publications. Books, articles, book chapters, and proceedings are published, whereas technical reports are made available in an informal manner, i.e. a publisher does not exist for this kind of publications. Keywords are used to suggest the subjects touched by publications. Several keywords may be assigned to a single publication and a single keyword may be assigned to multiple publications. The database keeps track of accesses to publications, keywords, and authors. Accesses from within the intranet are distinguished from remote accesses.

a. Specify SQL DDL statements to create user-defined structured types for the entities given in the conceptual schema! What variations of object identity exist? Specify SQL Reference Types for each alternative!

```
create type PublisherT as (
    name varchar(100),
    address varchar(100),
    url varchar(200)
)
not final
instantiable
ref using integer;

create type AccessCntT as (
    localAccesses integer,
    remoteAccesses integer
);

create type KeywordT as (
    topic varchar(50),
    accesscnt AccessCntT
)
ref using integer;
```

```
create type AuthorT as (
    name varchar(35),
    firstname varchar(25),
    accesscnt AccessCntT
) ref using integer;

create type PublicationT as (
    title varchar(150),
    url varchar(150),
    pdate date,
    accesscnt AccessCntT
) ref using varchar(20);

create type ContainedPublT under PublicationT as(
    source_title varchar(200),
    source_url varchar(300),
    first_page integer,
    last_page integer,
    publisher ref(PublisherT)
);

create type BookT under PublicationT as (
    publisher ref(PublisherT)
);

create type TechReportT under PublicationT as (
    number varchar(50),
    organisation varchar(200)
);

create type ArticleT under ContainedPublT as (
    volume integer,
    number integer
);
```

```
create type BookChapterT under ContainedPublT as (
    chapter varchar(15)
);


create type InProceedingsT under ContainedPublT as (
    location varchar(200)
);
```

b. Specify SQL DDL statements to define typed tables based on the user-defined structured types! Furthermore specify SQL DDL statements to define any necessary (untyped) tables!

```
create table publisher of PublisherT (
    ref is pid user generated,
    name with options NOT NULL
);

create table keyword of KeywordT (
    ref is kwid user generated,
    topic with options NOT NULL unique
);

create table author of AuthorT (
    ref is auid user generated,
    name with options not null,
    firstname with options not null,
    constraint author_uc unique (name, firstname)
);

create table Publication of PublicationT (
    ref is publKey user generated,
    title with options not null
);
```

```
create table ContainedPubl of ContainedPublT
    under publication;

create table Book of BookT under Publication (
    publisher with options not null scope Publisher
);

create table TechReport of TechReportT under Publication (
    number with options not null
);

create table Article of ArticleT under ContainedPubl;

create table BookChapter of BookChapterT
    under ContainedPubl;

create table InProceedings of InProceedingsT
    under ContainedPubl;

create table publkeywords (
    keyw ref(KeywordT) not null,
    publication ref(PublicationT) not null,
    constraint keywords_uc unique (keyw, publication)
);
alter table publkeywords
    alter publication add scope publication;
alter table publkeywords
    alter keyw add scope keyword;

create table publauthors (
    publication ref(PublicationT) not null,
    author ref(AuthorT) not null,
    position integer,
    constraint authors_uc unique (author, publication)
);
```

```
alter table publauthors
    alter publication add scope publication;
alter table publauthors
    alter author add scope author;
```

## 2) Querying Typed Tables

Specify SQL queries to retrieve the following information.

a. The name and address of all publishers.

```
select name, address
from publisher
```

b. All authors (name) whose first name starts with an “A”

```
select name
from author
where firstname like 'A%'
```

c. All articles (publkey) ordered by year of publication.

```
select publkey
from article
order by year(pdate)
```

d. All authors (name) and the number of their publications.

```
select author->name, count(*)
from publauthors
group by author
```

e. All authors (name) with the publications (title) they contributed to.

```
select author->name, publication->title
from publauthors;
```

f. The author with the highest number of publications.

```
select author->name
from publauthors
group by author
having count(*) =
   (select max(cnt) from
      (select count(*) as cnt
      from publauthors
      group by author) as maxcnt)
```

g. All authors (name) with the number of local accesses.

```
select name, accesscnt.localAccesses
from author
```

h. The sum of remote accesses for each author (name).

```
select author->name,
       sum(author->accesscnt.remoteAccesses)
from publauthors
group by author
```

i. Pairs of authors (names) that have more than one joint publication.

```
select a.author->name, b.author->name, count(*) as cnt
from publauthors a, publauthors b
where a.publication = b.publication and
      a.author <> b.author
group by a.author->name, b.author->name
having count(*) > 1
```

j. The name of the author with OID 0.

```
select name
from author
where auid = AuthorT(0)
```

```
select name
from author
where cast (auid as integer) = 0

select name
from author
where integer(auid) = 0
```

k. All publications (without its subtypes).

```
select title
from only(Publication)
```

l. All articles and books (title).

```
select title from Article
union all
select title from Book

select title
from Publication
where deref(publKey) is of (ArticleT) or
      deref(publKey) is of (BookT)
```

m. All contained publications (source_title, source_url, first_page, last_page, volume, number, chapter, location).

```
select *
from outer(ContainedPubl)
```

n. All publications with a flag to indicate whether they are contained in a collection or not (title, flag).

```
select publKey->title,
    (case when (deref(publKey) is of (ContainedPublT))
          then 1 else 0 end) as isContained
from Publication
```

o. The highest number of joint publications of two authors.

```
select max(cnt)
from (select a.author->name, b.author->name, count(*) as cnt
      from publauthors a, publauthors b
      where a.publication = b.publication and
            a.author <> b.author
      group by a.author->name, b.author->name
      having count(*) > 1) as rl
```

p. All publications of authors named “Smith” with a keyword that contains the word “object”.

```
select a.publication->title, k.keyw->topic
from publauthors a join publkeywords k on
     a.publication = k.publication
where a.author->name = 'Smith' and
      lcase(k.keyw->topic) like '%object%';
```

q. All publications with their keywords (if any, otherwise with null value).

```
select a.publication->title, a.author->name,
       k.keyw->topic, a.position
from publauthors a left outer join publkeywords k on
     a.publication = k.publication ;
```

r. All publications (title) that have been more often accessed remotely than locally.

```
select title
from publication
where accesscnt.localAccesses < accesscnt.remoteAccesses
```

s. Insert a new author named “John Doe” with OID 1.

```
insert into author
values(Authort(1), 'Doe', 'John', AccessCntt());
```

t. Increment the remote access counter of the publication with OID “XY2010a” by one.

```
update publication
set accessCnt.remoteAccesses = accessCnt.remoteAccesses + 1
where publkey = publicationT('XY2000b')
```