Data Definition, Relational Manipulation and Data Control Using SQL

Languages of DBMS

- Data Definition Language DDL
  - define the logical schema (relations, views etc) and storage schema stored in a Data Dictionary
- Data Manipulation Language DML
  - *Manipulative* populate schema, update database
  - *Retrieval* querying content of a database
- Data Control Language DCL
  - permissions, access control etc...

Data Definition:Creating tables

```
create table accountants
as(select studno, name, tutor, year
from student where hons = 'ca');
```

- Can specify column names, default values and integrity constraints (except referential)
- Datatypes and lengths derived from query
- Not null constraints passed on from query tables

Defining a Relation

```
create table student
(studentno number(8) primary key,
givenname char(20),
surname char(20),
hons char(3) check (hons in ('cis','cs','ca','pc','cm','mcs')),
tutorid number(4),
yearno number(1) not null,
constraint year_fk
  foreign key (yearno) references year(yearno),
constraint super_fk
  foreign key (tutorid) references staff(staffid));
```

Data Definition: Create Table

```
create table enrol
  (studno number(8),courseno char(5),
  primary key (studno, courseno),
  cluster (studno),
  labmark number(3)
     check (labmark between 0 and 100),
  exammark number(3)
     check (exammark between 0 and 100),
  constraint stud_fk
     foreign key (studno) references student,
  constraint course_fk
     foreign key (courseno) references course);
```

Data Definition: Altering Relations

```
alter table student
   add (address char(20),
          default null);

alter table student
   modify (name not null);
```

- this won't work if there are any nulls in the name column

Data Manipulation: Insert Operator

Course

courseno	subject	equip
cs250	prog	sun
cs150	prog	sun
cs260	graphics	sun
cs270	elec	pc
cs280	design	sun
cs290	specs	paper
cs390	specs	sun

```
insert into table
where search-condition
```

insert (cs310, elec, sun) into course;

```
insert into course (courseno,subject,equip)
      values ('cs310', 'elec', 'sun');
insert into course
           values ('cs310', 'elec', NULL);
```

Inserting Tuples into a Relation

```
insert into weak_students
   (studno,name,courseno,exammark)
where (select s.studno,name,courseno,exammark
   from enrol, student s
   where exammark <= 40 and
            enrol.studno = s.studno );
```

Insertion Anomalies

- An insert operation might voliate the uniqueness and minimality properties of the primary key of the referential integrity constraint
- insert (cs250,databases,sun) into course

COURSE

courseno	subject	equip
cs250	prog	sun
cs150	prog	sun
cs280	design	sun
cs290	specs	paper
cs390	specs	sun

Insertion anomalies can be corrected by
- rejecting the insertion
- correcting the reason for rejecting the update

Data Manipulation: Update Operator

```
update table
set column = expression
[where search-condition]
```

```
update enrol
set labmark = labmark * 1.1
where courseno = 'cs250';
```

- Modifies a tuple or tuples of a relation
- Don't violate constraints as long as the modified attributes are not primary keys or foreign keys
- Update of a primary key corresponds to a deletion followed by an insertion
- Update of a foreign key attribute is legal only if the new value corresponds to an existing tuple in the referenced relation or is null

Data Manipulation: Delete Operator

```
delete
from table
[where search-condition]
```

```
delete from course
where equip = 'pc';

delete from student
where year = '3' and(hons
!= 'mi' or hons <> ' si' );
```

- Deletes a tuple or a set of tuples from a relation
- Might violate the referential integrity constraint
- Anomalies can be overcome by
  - *rejecting* the deletion
  - *cascading* the deletion (delete tuples that reference deleted tuple)
  - *modifying* the referencing attribute values

Delete Operator

```
delete from student
where studno in
  (select student.studno
  from enrol e, teach t, student s
  where t.lecturer = 'woods'
  and t.courseno = e.courseno
  and e.studno = s.studno);
```

Data Control: Data Sharing and Security

- Permissions, access control etc...
- `create view` myyear `as`
  `select * from` student
  `where` year `in`
  `(select` year
  `from` student
  `where` name = `user)`
  `with check option`

Data Control: Data Sharing and Security

`grant` privilege, privilege2… | `all`
on table | view
`to` userID | roleID

`grant select` on student `to` bloggsf;

- Grant can be attached to any combination of `select, insert, update, delete, alter`
- Restricting access to parts pf a table can be effected by using the `view` and `grant` commands
- Privileges can be withdrawn with the `revoke` command

Synonyms for Objects

- `select name from` CAROLE.student;
- `create [public] synonym` synonym_name `for` table | view;
- `create synonym` student `for` CAROLE.student;
- `drop synonym` mystudent;

The Role of the Data Dictionary

- A set of tables and views to be used by the RDBMS as a reference guide to the data stored in the database files
- Every user retrieves data from views stored in the Data Dictionary
- The Data Dictionary stores:
  - user names of those permitted to access the database
  - names of tables, space definitions, views, indexes, clusters, synonyms etc
  - rights and privileges that have been granted