SQL Data Manipulation

Thomas Heinis

t.heinis@imperial.ac.uk
wp.doc.ic.ac.uk/theinis

Imperial College
London

SQL Data Manipulation

In addition to retrieving data from a databases, SQL provides support for data insertion, update and deletion.

The SQL statements that support these operations (as well as queries) are said to be a part of SQL's Data Manipulation Language (DML).

In contrast SQL's Data Definition Language (DDL) is concerned with schema creation and modification; as well as the specification of constraints and performance choices such as indexing.

Tuple Insertion

The SQL **insert** statement is used to add one or more tuples to a relation.

Example: movie(title, year, length, genre, studio, producer)

insert into movie(title, year, genre)
values (‘True Grit’, 2010, ‘Western’),
(‘The Kings Speech’, 2010, ‘Drama)

Missing attributes (e.g. length, studio, producer) are set to **null** (unless prohibited by the schema) or to a default value (if specified in the schema).

If **null** is prohibited and no default is specified in the schema then an error results and the tuple is not inserted into the relation. More generally if an insertion of a tuple violates a constraint for the relation, the insertion results in an error and causes rollback of any associated transaction.

Tuple insertion using a subquery

Rather than inserting individual tuples we can insert a set of tuples using a subquery.

Example: movie(title, year, length, genre, studio, producer)
studio(name, address, boss)

Adds studios in movie relation that are not already in studio relation

insert into studio(name)
select distinct studio **from** movies m
where m.studio **not in** (**select** s.name **from** studio s)

Note that an insert subquery is evaluated fully before insertion. This is to prevent any anomalies that might arise if tuples were inserted while the subquery was being evaluated.

Deletion

The SQL **delete** statement is used to remove one or more tuples from a relation that satisfy some condition.

Example: movie(title, year, length, genre, studio, producer)
casting(title, year, name)

```
delete from movie
where length > 180 or
            (title, year) in (select title, year from casting
  where name='Keanu Reeves')
```

Note: If the **delete** violates any database constraints then the deletion may not be done or may require further actions

If the **where** clause is omitted all tuples of the relation are deleted, leaving an empty relation!

Attribute updates

The SQL **update** statement is used to update one or more tuple attributes that satisfy a condition.

Example: **update** employee

set salary = **case**

when salary <= 70000 **then** salary * 1.02

when salary <= 80000 **then** salary * 1.03

else salary * 1.04

end,

lastpayincrease = current_date

where position = ‘Professor’

Again, updates are not performed if they violate the constraints for the relation.

Updates can use subqueries including correlated subqueries.

Example

Relations:

product(company, model, type)
- (‘Pear’, 2000, ‘desktop’)
- (‘Bell’, 1400, ‘laptop’)
- (‘HQ’, 500, ‘printer’)

desktop(model, price, speed, ram, hd)
- (2000, 450, 2.4, 4000, 500)

laptop(model, price, speed, ram, hd, screen)
- (1400, 550, 1.6, 2000, 250, 15)

printer(model, price, colour, type)
- (500, 120, True, ‘Inkjet’)

Example

Relations:

product(company, model, type)
- (‘Pear’, 2000, ‘desktop’)
- (‘Bell’, 1400, ‘laptop’)
- (‘HQ’, 500, ‘printer’)

desktop(model, price, speed, ram, hd)
- (2000, 450, 2.4, 4000, 500)

laptop(model, price, speed, ram, hd, screen)
- (1400, 550, 1.6, 2000, 250, 15)

printer(model, price, colour, type)
- (500, 120, True, ‘Inkjet’)

Q. Add desktop model 300 made by Pear, with speed 2.2 GHz, 4GB RAM, 300GB hard disk and price £495 to the database

Q. Insert into the database, the fact that for every laptop there is a desktop with the same company, speed, RAM size, hard disk size, with a model number 1000 more and a price £100 less.

Solution 1

Q. Add desktop model 300 made by Pear, with speed 2.2 GHz, ram 4GB, hard disk 300 GB and price £495 to the database.

```
insert into product(company, model, type)
values ('Pear', 300, 'desktop');
insert into desktop(model, price, speed, ram, hd)
values (300, 495, 2.2, 4000, 300)
```

Q. Insert into the database, the fact that for every laptop there is a desktop with the same company, speed, ram, hard disk, with a model number 1000 more and a price £100 less.

```
insert into product(company, model, type)
select company, model+1000, 'desktop' from product where type='laptop';
insert into desktop(model, price, speed, ram, hd)
select model+1000, price-100, speed, ram, hd from laptop;
```

Example 2

Relations:

product(company, model, type)
- (‘Pear’, 2000, ‘desktop’)
- (‘Bell’, 1400, ‘laptop’)
- (‘HQ’, 500, ‘printer’)

desktop(model, price, speed, ram, hd)
- (2000, 450, 2.4, 4000, 500)

laptop(model, price, speed, ram, hd, screen)
- (1400, 550, 1.6, 2000, 250, 15)

printer(model, price, colour, type)
- (500, 120, True, ‘Inkjet’)

Q. Pony buys DeePC. Reflect this change in the database.

Q. For each laptop made by Bell, add 1” to the screen size and subtract £50 from the price.

Solution 2

Q. Pony buys DeePC. Reflect this change in the database.

```
update product
set      company = 'Pony'
where    company = 'DeePC'
```

Q. For each laptop made by Bell, add 1” to the screen size and subtract £50 from the price.

```
update laptop
set      screen = screen + 1,
         price = price - 50
where    model in (select model from product where company='Bell')
```

Example 3

Relations:

product(company, model, type)
- (‘Pear’, 2000, ‘desktop’)
- (‘Bell’, 1400, ‘laptop’)
- (‘HQ’, 500, ‘printer’)

desktop(model, price, speed, ram, hd)
- (2000, 450, 2.4, 4000, 500)

laptop(model, price, speed, ram, hd, screen)
- (1400, 550, 1.6, 2000, 250, 15)

printer(model, price, colour, type)
- (500, 120, True, ‘Inkjet’)

Q. Delete all laptops made by companies that don’t make printers.

Solution 3

Q. Delete all laptops made by companies that don't make printers.

```
delete from laptop
where model in
        (select p1.model from product p1
         where p1.type='laptop' and p1.company not in
                (select p2.company from product p2
                 where p2.type = 'printer'
                )
        )
```

laptop models by companies that don't make printers

companies that make printers

In addition we should delete corresponding laptop tuples in product.

For readability we could also use views (see later) to name particular relational expressions, e.g. printer companies