Modification of the Database

part5

Modification of the Database

- Insert tuples in a relation
- Deletion of tuples from a given relation
- Insertion of new tuples into a given relation
- Updating values in some tuples in a given relation

Insert into ..values

- syntax
- INSERT INTO *table_name* (*column1*, *column2*, *column3*, ...)
  VALUES (*value1*, *value2*, *value3*, ...);

- INSERT INTO *table_name*
  VALUES (*value1*, *value2*, *value3*, ...);

Modification of the Database – Insertion

- Add a new tuple to *course*

  insert into *course*
  values (’CS-437’, ’Database Systems’, ’Comp. Sci.’, 4);

- or equivalently

  insert into *course* (*course_id*, *title*, *dept_name*, *credits*)
  values (’CS-437’, ’Database Systems’, ’Comp. Sci.’, 4);

- Add a new tuple to *student* with *tot_creds* set to null

  insert into *student*
  values (’3003’, ’Green’, ’Finance’, *null*);

Example table

- CREATE TABLE Persons (
  Customerid int NOT NULL AUTO_INCREMENT,
  CustomerName varchar(20),
- ContactName varchar(20),
- Address, City varchar(20),
- PostalCode int(10),
- Country varchar(20),
- PRIMARY KEY (Customerid)

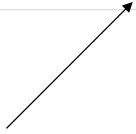
);

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
89	White Clover Markets	Karl Jablonski	305 - 14th Ave. S. Suite 3B	Seattle	98128	USA
90	Wilman Kala	Matti Karttunen	Keskuskatu 45	Helsinki	21240	Finland
91	Wolski	Zbyszek	ul. Filtrowa 68	Walla	01-012	Poland
92	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway

- INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)
  VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen , 21', 'Stavanger', '4006', 'Norway');

- Insert Data Only in Specified Columns
- INSERT INTO Customers (CustomerName, City, Country)
  VALUES

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
89	White Clover Markets	Karl Jablonski	305 - 14th Ave. S. Suite 3B	Seattle	98128	USA
90	Wilman Kala	Matti Karttunen	Keskuskatu 45	Helsinki	21240	Finland
91	Wolski	Zbyszek	ul. Filtrowa 68	Walla	01-012	Poland
92	Cardinal	null	null	Stavanger	null	Norway



Insertion (Cont.)

- Add all instructors to the *student* relation with tot_creds set to 0

student(SID, name, dept_name, tot_cred)

instructor(ID, name, dept_name, salary)

```
insert into student
        select ID, name, dept_name, 0
         from   instructor
```

- The **select from where** statement is evaluated fully before any of its results are inserted into the relation (otherwise queries like

```
insert into student select * from student
```

might insert an infinite number of tuples, if the primary key constraint on *student* were absent.)

deletion

- DELE

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
89	White Clover Markets	Karl Jablonski	305 - 14th Ave. S. Suite 3B	Seattle	98128	USA
90	Wilman Kala	Matti Karttunen	Keskuskatu 45	Helsinki	21240	Finland
91	Wolski	Zbyszek	ul. Filtrowa 68	Walla	01-012	Poland
92	Cardinal	Tom B. Erichsen	Skagen 21	Stavanger	4006	Norway

- DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';

Modification of the Database – Deletion

- Delete all instructors

  delete from *instructor*

- Delete all instructors from the Finance department

  delete from *instructor* **where** *dept_name*= 'Finance';

- Delete all tuples in the *instructor* relation for those instructors associated with a department located in the Watson building.

  delete from *instructor*
  where *dept_name* **in** (**select** *dept_name*
  from *department*
  where *building* = 'Watson');

Deletion (Cont.)

- Delete all instructors whose salary is less than the average salary of instructors

```
delete from instructor
where salary< (select avg (salary) from instructor);
```

updation

- The UPDATE statement is used to modify the existing records in a table.

- UPDATE *table_name*
  SET *column1* = *value1*, *column2* = *value2*, ...
  WHERE *condition*;

- UPDATE Customers SET ContactName = 'Alfred Schmidt', City= 'Frankfurt‘
  WHERE CustomerID = 1;

- UPDATE Customers SET ContactName='Juan';

Modification of the Database – Updates

- Increase salaries of instructors whose salary is over $100,000 by 3%, and all others receive a 5% raise
  - Write two **update** statements:

```
update instructor
  set salary = salary * 1.03
  where salary > 100000;
```

```
update instructor
    set salary = salary * 1.05
    where salary <= 100000;
```

  - **The order is important**
  - Can be done better using the **case** statement (next slide)

Case Statement for Conditional Updates

- Same query as before but with case statement

```
update instructor
set salary = case
        when salary <= 100000 then salary *
1.05
        else salary * 1.03
        end
```

The general form of the case statement is as follows.

```
CASE
    WHEN condition1 THEN result1
    WHEN condition2 THEN result2
    WHEN conditionN THEN resultN
    ELSE result
END;
```

- The following SQL will order the customers by City. However, if City is NULL, then order by Country:

- ```
  SELECT CustomerName, City, Country
  FROM Customers
  ORDER BY
  (CASE
      WHEN City IS NULL THEN Country
      ELSE City
  END);
  ```
  ```

Updates with Scalar Subqueries

- Recompute and update tot_creds value for all students
set the *tot cred* attribute of each *student* tuple to the **sum of the credits** of courses successfully completed by the student

```
update student S
   set tot_cred = ( select sum(credits)
                    from takes natural join course
                    where S.ID= takes.ID and
                          takes.grade <> 'F' and
                          takes.grade is not null);
```

End of Chapter 3