Politecnico
di Torino

DBMG

Update commands

SQL language: basics

SQL language: update commands

- Introduction
- The INSERT command
- The DELETE command
- The UPDATE command

Update instructions

- Update operations alter the state of the database
  - integrity constraints must be checked to ensure that they are still verified
- Each instruction may update the contents of a single table

- INSERT
  - inserting new tuples into a table
- DELETE
  - deleting tuples from a table
- UPDATE
  - modifying the content of tuples in a table

INSERT

- Inserting a single tuple
  - assignment of a constant value to each attribute

    INSERT INTO *TableName*

    [(ColumnList)]

    VALUES *(CostantList);*

- Inserting multiple tuples
  - read from other tables by means of a SELECT command
  - it must not include an ORDER BY clause

    INSERT INTO *TableName*

    [(ColumnList)]

    Query;

Example 1: Inserting a tuple

- Insert product P7 with Name: Jumper, Color: Purple, Size: 40, Store: Helsinki

```
INSERT INTO P (PId, PName, Color, Size, City)
        VALUES ('P7', 'Jumper', 'Purple',40,'Helsinki');
```

- A new tuple is inserted into table P with the specified values
- Omitting the field list is equivalent to specifying all fields, according to the column order specified upon table creation
  - If the table schema changes, the INSERT command is no longer valid

Example 2: Inserting a tuple

- Insert product P8 with Store: Istanbul, Size: 42

```
INSERT INTO P (PId, Store, Size)
        VALUES ('P8', 'Instanbul', 42);
```

- A new tuple is inserted into table P with the specified values
  - PName and Color are assigned the NULL value
- For all attributes whose values are not specified, the domain of the attribute must allow the NULL value

Example 3: Referential integrity constraints

- Insert a new supply for supplier S20, product P20 and quantity 1000

```
INSERT INTO SP (SId, PId, Qty)
        VALUES (‘S20', 'P20', 1000);
```

- Referential integrity constraint
  - P20 and S20 must already be present in the P and S tables, respectively
  - if the constraint is not satisfied, the insertion should not be executed

Example 4: Inserting multiple records

TOTAL-SUPPLIES (PId, TotalQty)

- For each product, insert the overall supplied quantity into table TOTAL-SUPPLIES
  - aggregate data extracted from table SP

```
INSERT INTO TOTAL-SUPPLIES (PId, TotalQty)
    (SELECT PId, SUM(Qty)
       FROM SP
       GROUP BY PId);
```

DELETE

DELETE FROM *TableName*
[WHERE predicate];

- Deletion of all tuples satisfying the predicate from table *TableName*
- It must be ensured that the deletion does not cause the violation of referential integrity constraints

Example 1: Clearing Table Contents

- Delete all supplies

```
DELETE FROM SP;
```

- If no WHERE clause is specified, all tuples satisfy the selection predicate
  - the contents of table SP are deleted
  - the table itself is *not* deleted

Example 2: Referential integrity contraints

- Delete the tuple corresponding to the supplier with code S1

```
DELETE FROM S
WHERE SId='S1';
```

- If SP includes supplies related to the deleted suppliers, the database loses its integrity
  - a violation of the referential integrity constraint between SP and S occurs
  - the deletion must be propagated

Example 2: Referential Integrity constraints

- Delete the tuple corresponding to the supplier with code S1

```
DELETE FROM S
WHERE SId='S1';

DELETE FROM SP
WHERE SId='S1';
```

- To maintain integrity, the deletion operations must be completed on both tables

Example 3: Referential integrity constraints

- Delete the suppliers based in Paris
- If SP includes supplies referring to the deleted suppliers, the referential integrity constraint between SP and S is violated
  - such supplies must also be deleted from SP

```
DELETE FROM SP
WHERE SId IN (SELECT SId
              FROM S
               WHERE City='Paris');

DELETE FROM S
WHERE City='Paris';
```

UPDATE

```
UPDATE TableName
SET column = expression
      {, column=expression}
[ WHERE predicate];
```

- All records in table *TableName* satisfying the predicate are modified according to the assignment *column=expression* in the SET clause

Example 1: Updating a tuple

- Update the features of product P1: assign Yellow to Color, increase the size by 2 and assign NULL to Store

```
UPDATE P
SET Color = 'Yellow',
      Size=Size+2,
      Store = NULL
WHERE PId='P1';
```

- The tuple identified by code P1 is updated

Example 2: Multiple updates

- Update all suppliers based in Paris by doubling the number of employees

```
UPDATE S
SET #Employees=2*#Employees
WHERE City='Paris';
```

- All tuples selected by the predicate in the WHERE clause are updated

Example 3: Update with nested query

- Update to 10 the quantity of supplied products for all suppliers based in Paris

```
UPDATE SP
SET Qty = 10
WHERE SId IN (SELECT SId
        FROM S
          WHERE City='Paris');
```

Example 4: Updating multiple tables

- Change the code of supplier S2 to S9

```
UPDATE S
SET SId='S9'
WHERE SId='S2';
```

- If SP includes supplies related to the updated suppliers, the referential integrity constraint is violated
  - such supplies must also be updated in SP

Example 4: Updating multiple tables

- Change the code of supplier S2 to S9

```
UPDATE S
SET SId='S9'
WHERE SId='S2';

UPDATE SP
SET SId='S9'
WHERE SId='S2';
```

- To maintain integrity, the update must be completed on both tables (integrity constraints checking must be temporarily disabled)