SQL
Elmasri/Navathe ch 4
Padron-McCarthy/Risch ch 7/8/9

DATABASE DESIGN I - 1DL300 spring 2013
Sobhan badiozamany
Department of Information Technology, Uppsala University

Structured Query Language

Note: This is an introduction in form of a tutorial. For the course&labs more SQL knowledge is needed. Official slides from the book will be available at Studentportalen.
SQL
Elmasri/Navathe ch 4
Padron-McCarthy/Risch ch 7/8/9
DATABASE DESIGN I - 1DL300 spring 2013
Sobhan badiozamany
Department of Information Technology, Uppsala University

Note:
This is an introduction in form of a tutorial. For the course&labs more SQL knowledge is needed. Official slides from the book will be available at Studentportalen.
Structured Query Language (SQL) is a standard database management language that is used to access and manipulate data in a relational database. It is divided into two main categories:

1. **Data Definition Language (DDL)**: This subset of SQL is used in creating, changing, and deleting the description (schema) of tables.

2. **Data Manipulation Language (DML)**: This subset of SQL is used in adding, modifying, and removing rows in the tables.

**Note:**
This is an introduction in form of a tutorial. For the course & labs more SQL knowledge is needed. Official slides from the book will be available at Studentportalen.
Data Definition Language

The subset of SQL that is used in creating, changing and deleting the description (schema) of tables.
Specifies a new table by giving it a name and specifying its columns and the data type (domain) of the columns.

```sql
CREATE TABLE DEPARTMENT(
    DNAME  VARCHAR(10) NOT NULL,
    DNUMBER  INTEGER NOT NULL,
    MGRSSN CHAR(9),
    PRIMARY KEY (DNUMBER),
    MGRSTARTDATE CHAR(9)
);
```

Let's try it!
The entity relationship diagram represents a database schema for a supply chain management system. Entities include:

- **Employee**: Name, Salary, Birthdate, Startdate, Manager (number)
- **Dept**: Name, Floor, Manager (number)
- **Subdivision**: Manager (number)
- **Store**: Number, Location (city, state)
- **Supplier**: Number, Location (city, state)
- **Parts**: Number, Name, Color, Weight, QOH
- **Debit**: Number, Sdate, Account
- **Sale**: Number, Quantity
- **Item**: Name, Price, Deliver
- **Supply**: Quantity, Shipdate

**Relationships**:
- Employee is the head_of relation for Dept
- Dept is the manages relation for Employee
- Dept is the subdivision relation for Subdivision
- Store is the location relation for City
- Supplier is the location relation for City
- Debit is the sale relation for Sale
- Item is the deliver relation for Sale
- Supply is the quan relation for Item

An example SQL command for creating a database table for Supplier could be:

```sql
CREATE TABLE Supplier (
    number INT,
    name VARCHAR,
    color VARCHAR,
    weight DECIMAL,
    qoh INT,
    PRIMARY KEY (number)
);```
The company decides to expand internationally . . .

What is the cardinality ratio of "Belongs to" relationship?

Let's create a table for the "Country" entity . . .

An example:
creating a department table.

```
CREATE TABLE DEPARTMENT(
  DNAME VARCHAR(10) NOT NULL,
  DNUMBER INTEGER NOT NULL,
  MGRSSN CHAR(9),
  PRIMARY KEY (DNUMBER),
  MGRSTARTDATE CHAR(9)
);
```
Let's create a table for the "Country" entity ...

An example: creating a department table.

```
CREATE TABLE DEPARTMENT(
    DNAME       VARCHAR(10) NOT NULL,
    DNUMBER     INTEGER NOT NULL,
    MGRSSN      CHAR(9),
    PRIMARY KEY (DNUMBER),
    MGRSTARTDATE CHAR(9)
);
```
Alter table

Is used to change definition of tables, such as adding/removing:
- attributes
- foreign keys.
Adding a job title attribute to the employee table:

ALTER TABLE EMPLOYEE
ADD JOB_TITLE VARCHAR(12);
Adding the foreign key to dept:
ALTER TABLE dept ADD CONSTRAINT fk_dept_store
FOREIGN KEY (store) REFERENCES store (number);

Dropping the same foreign key from dept:
ALTER TABLE dept
DROP FOREIGN KEY fk_dept_store;
Alter table

Is used to change definition of tables, such as adding/removing:
- attributes
- foreign keys.
The company decides to expand internationally . . .

How should we implement the "Belongs to" relationship?

A side note: A common mistake is to draw foreign keys as [oval shaped] attributes in ER. The relationship (here "Belongs to") represents the concept of foreign key.

To implement (N-1) relationships, we need a foreign key attribute in the entity that is on the N side, here being "city".
Add a "country" attribute to city, similar to:

```
ALTER TABLE [table_name] ADD [Attribute_X] [Data type];
```

What should be the data type of this "country" attribute?
The company decides to expand internationally . . .

**How should we implement the "Belongs to" relationship?**

To implement (N-1) relationships, we need a foreign key attribute in the entity that is on the N side, here being "city".

A side note:
A common mistake is to draw foreign keys as [oval shaped] attributes in ER. The relationship (here "Belongs to") represents the concept of foreign key.
Define a foreign key constraint that enforces the relationship, similar to:

```
ALTER TABLE [table_name]
ADD CONSTRAINT [fk_name] FOREIGN KEY ([fk_attribute])
REFERENCES [referenced table] ([pk_attribute]);
```
A common mistake is to draw foreign keys as [oval shaped] attributes in ER. The relationship (here "Belongs to") represents the concept of foreign key.

A side note:

How should we treat the relationship in the ER model?
Drop table

Removes the definition of table from the database schema, which means:
- The table will no longer be accessible.
- Data in the table would be lost.

```sql
DROP TABLE table_name;
```
Data Definition Language

The subset of SQL that is used in creating, changing and deleting the description (schema) of tables.
Data Manipulation Language

The subset of SQL used in adding, modifying and removing rows in the tables.
**Insert**

Used in inserting rows into a table.

**Inserting by specifying a row**

```
NAME | MA | SEX | MK | 00 | NV | 00 | 00 | 00 | 00 | 00 |
INSERT INTO EMPLOYEE VALUES
(Richard, Y, Martin; 6525948533, 78-00-50-32; 19 Oak
Forest City, TX, 75000; 00/00/2013; 23.; 2.5).
```

- The order of attributes values should match the order of attributes in CREATE TABLE command.

```
INSERT INTO EMPLOYEE (ENAME, LNAMw, SMN)
VALUES (Richard, Martin, 6525948533, 78-00-50-32;)
```

- The order of values should match the order specified within the command.
- NULL will be assigned to unspecified attributes.

**Inserting results of a query**

Suppose we want to copy data from one table to another...

```
SELECT ENAME, COUNT(*), SUM(SALARY)
FROM DEPARTMENT, EMPLOYEE
WHERE E NUMBER = D NO
GROUP BY ENAME;
```

Then insert the result into the destination table.

```
INSERT INTO DEPTS_INFO
(DEPT_NAME, NO_OF_EMPS, TOTAL_SAL)
```

First, write a query that produces the rows that you are going to insert.
Inserting by specifying a row

**EMPLOYEE**

<table>
<thead>
<tr>
<th>FNAME</th>
<th>MINIT</th>
<th>LNAME</th>
<th>SSN</th>
<th>BDATE</th>
<th>ADDRESS</th>
<th>SEX</th>
<th>SALARY</th>
<th>SUPERSSN</th>
<th>DNO</th>
</tr>
</thead>
</table>

**INSERT INTO EMPLOYEE VALUES**
('Richard','K','Marini', 653298653, '30-DEC-52','98 Oak Forest,Katy,TX', 'M', 37000,987654321, 4);

- The order of attribute values should match the order of attributes in create table command.

**INSERT INTO EMPLOYEE (FNAME, LNAME, SSN) VALUES** ('Richard', 'Marini', 653298653);

- Here the order of values should match the order specified within the command.
- NULL will be assigned to unspecified attributes.

What happens if Null value is not allowed for an unspecified attribute?
Inserting by specifying a row

<table>
<thead>
<tr>
<th>EMPLOYEE</th>
</tr>
</thead>
<tbody>
<tr>
<td>FNAME</td>
</tr>
</tbody>
</table>

**INSERT INTO EMPLOYEE VALUES**

('Richard', 'K', 'Marini', 653298653, '30-DEC-52', '98 Oak Forest, Katy, TX', 'M', 37000, 987654321, 4);

- The order of attribute values should match the order of attributes in create table command.

**INSERT INTO EMPLOYEE (FNAME, LNAME, SSN) VALUES**

('Richard', 'Marini', 653298653);

- Here the order of values should match the order specified within the command.
- NULL will be assigned to unspecified attributes.

**What happens if Null value is not allowed for an unspecified attribute?**
Now let's insert two countries into our newly created "country" table:
- USA with population of 314,000,000, an English speaking country
- Sweden with population of 9,000,000 a Swedish speaking country

Then insert two Swedish cities in the city table:
- Uppsala
- Stockholm

Recall that the city table has three attributes:
- Name
- State
- Country

An example insert command:

```
INSERT INTO EMPLOYEE (FNAME, LNAME, SSN) VALUES ('Richard', 'Marini', 653298653);
```
Now let's see the contents of the country and city tables:

\texttt{Select * from country;}

\texttt{Select * from city;}

`
Inserting results of a query

Suppose we want to copy data from one table to another . . .

```
INSERT INTO DEPTS_INFO
(DEPT_NAME, NO_OF_EMPS, TOTAL_SAL)
SELECT
    DNAME, COUNT (*), SUM(SALARY)
FROM
    DEPARTMENT, EMPLOYEE
WHERE
    DNUMBER = DNO
GROUP BY
    DNAME;
```

Then insert the result into the destination table

First, Write a query that produces the row(s) that you are going to insert.
INSERT INTO DEPTS_INFO
(DEPT_NAME, NO_OF_EMPS, TOTAL_SAL)
SELECT
  DNAME, COUNT (*), SUM(SALARY)
FROM
  DEPARTMENT, EMPLOYEE
WHERE
  DNUMBER = DNO
GROUP BY
  DNAME;

Then insert the result into the destination table.

First, write a query that produces the row(s) that you are going to insert.
Suppose we want to copy data from one table to another . . .

```
INSERT INTO DEPTS_INFO
    (DEPT_NAME, NO_OF_EMPS, TOTAL_SAL)
SELECT
    DNAME, COUNT (*), SUM(SALARY)
FROM
    DEPARTMENT, EMPLOYEE
WHERE
    DNUMBER = DNO
GROUP BY
    DNAME;
```

Then insert the result into the destination table.

First, write a query that produces the row(s) that you are going to insert.
Update

Used to modify attribute values of one or more selected rows.

Update [table name] set ... where ...

Referential integrity

- Integrity constraints can refer to domain or table operations.
- Updating an attribute value to a value that is not in the domain will not be allowed.
- Inserting a row into a table with a referenced column that is not in the domain.
- Updating the value of a foreign key attribute to a value that does not exist in the referencing table will not be allowed.

- Updating a country column in the city table in a foreign entity.
SET-clause specifies the attributes to be modified and their new values

SET Salary = 1000, manager = "John"
The where clause selects the rows to be modified.

WHERE city="Stockholm"
Referential integrity

*Integrity constraints are enforced during update operations*

Updating an attribute value to a value that is not in the domain will not be allowed.

- updating population of Sweden to "nine million"

Updating the value of a foreign key attribute to a value that does not exist in the referencing table will not be allowed.

- Updating country column in the city table to a none existing country.
Update city and country data

1. Change the population of US to 314,500,000

2. In addition, in the recent year, there has been 1% addition to the population of all countries, apply that.

3. By looking at the rows in the city table, American cities have NULL value for the field country, update all of them to "USA".

   The update command looks like:

   Update country
   set ...
   where ...

   At any point you can see the contents of the country and city tables by:

   Select * from country;
   Select * from city;
NULL is a very special value!

Each NULL value distinct from other NULL values

Therefore, equality comparison is not appropriate

SQL uses "IS" or "IS NOT" to compare NULLs.
Update city and country data

1. Change the population of US to 314,500,000

2. In addition, in the recent year, there has been 1% addition to the population of all countries, apply that.

3. By looking at the rows in the city table, American cities have NULL value for the field country, update all of them to "USA".

The update command looks like:

```
Update country
set ...
where ...
```

At any point you can see the contents of the country and city tables by:

```
Select * from country;
Select * from city;
```
Delete

Removes rows from a table.

DELETE FROM [table name]
WHERE ...

- The `WHERE` clause selects the rows to be deleted.
- If no `WHERE` clause is specified, all rows will be deleted.

Example:
```
DELETE FROM Employees WHERE country = 'Sweden';
```
The where clause selects the rows to be deleted.

WHERE country="Sweden"

if no where clause is specified --> all rows will be deleted
If we remove a country, what happens to its cities?

Depends on the definition of foreign key:
- If "on delete cascade" is specified, all cities will be removed too.
- Otherwise, the cities will have Null value for country.
Structured Query Language

Data Definition Language

The subset of SQL that is used in creating, changing, and deleting the description (schema) of tables.

Queries

Data Manipulation Language

The subset of SQL used in adding, modifying, and removing rows in the tables.

Note:
This is an introduction in form of a tutorial. For the course & labs more SQL knowledge is needed. Official slides from the book will be available at Studentportalen.
Structured Query Language

Data Definition Language
The subset of SQL that is used in creating, changing, and deleting the description (schema) of tables.

Data Manipulation Language
The subset of SQL used in adding, modifying and removing rows in the tables.

Queries

Note:
This is an introduction in form of a tutorial. For the course & labs more SQL knowledge is needed. Official slides from the book will be available at Studentportalen.