Geographic Information Technology Training Alliance (GITTA) presents:

Structured Query Language SQL

Responsible persons: Anca Dobre, Dominique Schneuwly, Susanne Bleisch

Table Of Content

1. Structured Query Language SQL

SQL (Structured Query Language) is a query language for relational databases. The roots of SQL go back to SQUARE, a more mathematical oriented language and SEQUEL, a predecessor of SQL from the seventies. There exist different standards (ISO and ANSI) of SQL, but the most common one is SQL-92. The latest standard SQL-99 (or SQL3) even includes XML. In the following Units we will give a simple overview of the most important SQL functions.

1.1. SQL Overview

SQL (Structured Query Language) is one of the main reasons for the commercial success of relational databases. The ANSI (American National Standards Institute) and the ISO (International Standards Organization) developed in 1986 the first SQL-version with the name SQL-86 or SQL1. In 1992 a second and more extended standard with the name of SQL-92 or SQL2 was established. The latest standard includes XML, dates from 1999 and is therefore called SQL-99 or SQL3. With the use of SQL in most commercial database systems the migration from one system to another has become easier for the user. In the ideal case, the user need not consider which system is used because query formation in SQL remains the same.

1.1.1. SQL Concepts

SQL is a descriptive, entity-oriented query language for data manipulation with its roots in relational algebra. Today SQL is used either as a stand-alone programming language or within other languages like C, C++, Java, ADA, COBOL, FORTRAN, PL/1, PASCAL etc.
SQL actually consists of three sub languages:

- DDL - Data Definition Language: Used for creating databases and tables and for maintaining the structure.
- DML - Data Manipulation Language: Used for accessing and extracting data from a database (add, update, delete etc.).
- DCL - Data Control Language: Used to control access to the database and therefore essential for the security system.

In most implementations of SQL functions from other programming languages (if-clauses, iterations etc.) have been added. Some SQL-versions, such as Oracle's PL/SQL, can therefore be seen as independent programming languages.

1.1.2. Data Definition (DDL)

In SQL the terms table, row and column are synonyms for relation, tuple and attribute. To create a new relation scheme in the database we start with the `create table`-command. Together with the creation we must provide the relations attributes and their domains (eg. number, char or date). Additional we can define other constraints, checks and keys etc. The keyword `NOT NULL` tells the system that this attribute cannot be empty. Primary keys are declared using a special `"table constraint"-clause`.

```
create table Subscription (
   Name char (30) not null,
   CustID number (5) not null,
   SubType char (15) default 'monthly' check (SubType in ('monthly', 'weekly')),
   SubStart date,
       constraint pk_sub primary key  (Name, CustID),
       constraint fk_name foreign key (Name) references Magazine (Name),
       constraint fk_custid foreign key  (CustID) references Customer (CustID));
```

Code-Beispiel: Create Table

This example shows how a new table is created in SQL. In the intermediate lesson we take a closer look at all the SQL statements. For the moment you don't have to understand these statements in detail.

1.1.3. Data Manipulation (DML)

There are two sorts of data manipulation commands in SQL. The first type are only used for database queries and do not alter the tables. The second type are used for adding, updating or deleting values in a database.
We will take a closer look at the different data manipulation commands in the following units.

- **Basic database queries**
- **SQL Insert, Delete and Update**

1.1.4. Data control (DCL)

Data control commands in SQL control access privileges and security issues of a database system or parts of it. These commands are closely related to the DBMS (Database Management System) and can therefore vary in different SQL implementations.
Some typical commands are:

- `GRANT` - give user access privileges to a database
- `DENY` - deny user access
- `REVOKE` withdraws access privileges given with the GRANT or taken with the DENY command

Since these commands depend on the actual database management system (DBMS), we will not cover DCL in this module.

1.2. Basic database queries

In this Unit we will take a closer look at how to do database queries using SQL.

1.2.1. SELECT-FROM-WHERE clause

In languages like SQL data from a certain domain (FROM) that match some conditions (WHERE) are selected and presented (SELECT). The result can be seen as a new relation.
The syntax of a basic SQL query is:

```
SELECT <select-list>
FROM <from-list>
WHERE <condition>
```

In this syntax the...

- <select-list> contains the names of the attributes (columns) values to be returned.
- <from-list> is the list of the relations (tables) used for the query.
- <condition> is an expression that identifies the tuples that we are looking for.

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

We saw the basic statements that are needed for making queries with SQL. Of course there are extensions which allow more specific or more flexible queries.
These extensions include:

- multiple conditions (in boolean AND/OR combination)
- complex conditions (subqueries, joins etc.)
- pattern matching and arithmetical operators
- non-relational functions (sort, group, aggregate)
- set operators (union, intersect, minus)

1.2.2. Multiple conditions

In SQL it is possible to have multiple conditions and combine them with boolean operators (AND, OR). For negating a condition the NOT operator is used.

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

1.2.3. Complex conditions

Nested queries
Conditions are usually made of an attribute, a value and an operator that forms the condition (eg. Name="John"). But the values themselves don't have to be constants, they can be the result of another sub-query. We then talk about nested queries. Using the IN-operator nested queries can be as deep as necessary.

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

Comparison operators
SQL supports different comparison operators like `==`, `<`, `>`, `<>`, `<=`, `>=`, `between` etc. If an operator is used to compare an attribute with a constant, we talk about restriction. If the operator is used to compare two attributes, then we talk about a join. With joins, data from different relations can be compared and combined. Of course only attributes with the same domain (value range) can be compared.

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

If in the SELECT-clause a star (*) instead of an attribute list is used, then all attributes are displayed.

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

1.2.4. Pattern matching and arithmetical operators

Pattern matching using LIKE
The LIKE condition allows you to use wildcards in the WHERE clause of an SQL statement. This allows pattern matching.
The patterns that you can choose from are:

- "%" allows you to match any string of any length (including zero length)
- "_" allows you to match on a single character

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

Arithmetical operators
The arithmetical standard operators for addition (+), subtraction (-), multiplication (*) and division (/) can all be used for numerical constant or for attributes with a numerical domain.

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

In the above example the result is a relation with three attributes. The third attribute is named "Usury" since "Price/Size" is not a good name and stands for the price per unit. We define that if it lies over 15, it is overpriced ("Usury").

1.2.5. Non-relational constructs

ORDER BY clause
SQL contains some operators that have nothing to do with relational algebra. For example an entity per definition does not have an order. Nevertheless in SQL you can order your tables using the ORDER BY clause. Using the keywords ASC and DESC the sorting can either be ascending or descending.

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

In this example all customers are sorted in ascending order according to their names and as a second sort parameter in descending order according to their surname. The ASC keyword is default and could therefore be omitted.

GROUP BY clause
Grouping methods are used to form a subset of tuples of a relation according to certain criteria. These subsets can then be used to calculate statistical parameters such as average, sum etc. A certain value of an attribute serves a grouping criteria.
The group functions that SQL usually offers are the following:

- `min` returns the smallest value ignoring null values
- `max` returns the largest value ignoring null values
- `sum` returns the sum of all values ignoring null values
- `count` returns the number of rows
- `avg` returns average value ignoring null values
- `stdev` returns the standard deviation ignoring null values
- `varriance` returns the variance ignoring null values

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

In this query we want to find the customers that spent more than 250 SFR for all their small advertisings (a small add is an add that costs less than 300 SFR). In a first step A004 is sorted out because it is not a small add (price is over 300 SFR). The remaining tuples are grouped by customer and those with a sum of over 250 SFR are selected.

1.2.6. Set operators

Set operators are used to connect different queries and produce a resulting relation. Of course these set operations are only allowed if the attributes match (eg. the domain or the number of attributes etc.). The following set operators are used to join together sets of tuples. They might already be known from the algebra of sets.

- `union` produces a union of different sets
- `intersect` produces an intersection of different sets.
- `minus` subtracts one set from another one.

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

This query is used to find all names of customers who have a magazine subscription and who have placed an advertising in this magazine.

1.2.7. Usage of SQL

This flash-animation (click on link) should allow you to exercise all kinds of database queries.
To start the animation, klick on the graphic

The following scheme is used:

Appartment

Landlord	Tenant	Rent
Schulze	Andreas	1100
Schulze	Nicole	2400
Schulze	Tanja	900

Community

Host	Guest	Date
Tanja	Markus	30.8.1998
Frank	Simone	31.8.1998
Rainer	Birgit	4.9.1998
Frank	Simone	4.9.1998
Nicole	Wolfgang	4.9.1998
Nicole	Simone	4.9.1998
Nicole	Andreas	5.9.1998
Katja	Birgit	5.9.1998

Visit

Main tenant	Subtenant	Rent
Andreas	Simone	600
Nicole	Katja	500
Nicole	Markus	500
Tanja	Birgit	450
Nicole	Frank	450
Andreas	Wolfgang	550
Nicole	Rainer	500

Flash-Animation

1.2.8. atabase queries

This self assessment should give you the possibility to test different SQL queries and get to know them more in depth. We use the following **sample database** (please leave this overview open during the exercise).
Find the correct solution tables for the following SQL queries. Put your solutions (either Word or PDF format) on the discussion board and check the other solutions.
You are encouraged to comment on other postings or to ask questions using the discussion board. Please do not email the tutor directly so that others can also benefit from your questions and the tutoräs answer.

SQL Anfragen

- ```
  SELECT vname, nname
  FROM angestellter
  WHERE salaer = 25000
  AND ahvnr > 5000
  ```
- ```
  SELECT pname
  FROM projekt
  WHERE abt = 5
  OR pnummer = 30
  ```
- ```
  SELECT nname
  FROM angestellter
  WHERE salaer BETWEEN 28000 AND 41000
  ```
- ```
  SELECT ahvnr
  FROM angestellter
  WHERE adresse LIKE 'Zu%'
  ```

- ```
  SELECT Angestellter.nname
  FROM Angestellter, Arbeitet_an
  WHERE Arbeitet_an.projekt = '20'
  AND Angestellter.ahvnr = Arbeitet_an.ang
  ```
- ```
  SELECT name
  FROM angehoeriger
  WHERE ang IN
  (SELECT ahvnr
  FROM angestellter
  WHERE adresse = 'Kuesnacht')
  ```

Some remarks about the WebSQL interface

- Do not use a semicolon at the end of a SQL query.
- On the bottom left you can reset the database to its original state.
- Please contact your tutor and not the IFI for technical questions.

Aufgaben

- Select the names of all employees and sort them in descending order (Z to A).
- Find the surnames of all employees who live in Dübendorf and earn more than 30'000 Fr.
- Select all children whose parents are employees and live in Zurich. Sort the result by birth date.
- Find the total amount of time that is spent on all projects in the research (Forschung) department. Use one query only!
- How many children has the manager of the administration (Verwaltung) department?
- Find projects of Zurich where the total amount of work is over 50 hours?
- Create your own SQL query and post the query together with the solution on the discussion board.
- Solve some queries that other students have posted.

1.3. SQL Insert, Delete and Update

To keep a database accurate we have to be able to not only create and delete tables but to also modify the content. In SQL this is done with the commands INSERT, DELETE and UPDATE.

1.3.1. Inserting tuples

In its most basic form the insert-command adds a tuple to an existing table.
The syntax is:
`INSERT INTO` Tablename (Attributelist)
`VALUES` (Valuelist);
Please note that the attributelist can be omitted if a complete tuple is inserted.

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

1.3.2. Deleting tuples

The delete-command removes one or more rows (tuples) of a table (relation). The WHERE-clause specifies which tuples have to be deleted. If it is missing, all tuples are deleted!
The syntax is:
`DELETE FROM` Tablename
`WHERE` Condition;

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

1.3.3. Updating tuples

The update-command is used to change one or more attribute values of existing tuples. The WHERE-clause specifies which tuples should be updated. The SET-clause specifies the new values.
The syntax is:
`UPDATE` Tablename
`SET` Attributevalues
`WHERE` Condition;

Only pictures can be viewed in this version! For Flash, animations, movies etc. see online version. Only screenshots of animations will be displayed. [link]

1.4. Summary

SQL stands for "Structured Query Language" and is a language to communicate with relational and object oriented databases. With SQL new tables (relations, schemes) can be created, altered and deleted using the commands `CREATE TABLE`, `ALTERTABLE`and `DROP TABLE`. This part of SQL is known as the Data Definition Language (DML). More important in the daily use of SQL are the data query and manipulation (DML) commands. These commands allow you to `INSERT, DELETE and UPDATE` values in the database. SQL is also used to control access restrictions to the database or to parts of it.

1.5. Recommended Reading

- **ELMASRI, R., NAVATHE, S.B.**, 1994. *Fundamentals of Database Systems*. 2nd. Redwood City, California: Addison-Wesley.
  Introduction to databases and SQL.

1.6. Bibliography

- **ELMASRI, R., NAVATHE, S.B.**, 1994. *Fundamentals of Database Systems*. 2nd. Redwood City, California: Addison-Wesley.