CSIT115 Data Management and Security

CSIT882 Data Management Systems

Introduction to SQL

Subject coordinators: Dr Chen Chen, Dr Thanh Le Hoang

Slides created by Dr Janusz R. Getta, School of Computing and Information Technology - University of Wollongong

Introduction to SQL

Outline

- Structured Query Language
- Characteristics
- Functionality
- Formatting

Structured Query Language

Defined and implemented by IBM in early 1970s

Originally called SEQUEL (Structured English QUEry Language)

First implementation: IBM's SYSTEM R (DB/2, UDB)

The first ANSI and ISO standard in 1986 (SQL-86)

The revisions in 1989, 1992, 1999, 2003, 2006, 2008, 2011 and 2016

SQL is a command oriented, declarative, common for all relational database management system database programming language

Introduction to Structured Query Laguage (SQL)

Outline

Structured Query Language

Characteristics

Functionality

Formatting

Characteristics

SQL is commonly used to:

- (1) Create databases and the objects within them
- (2) Store data in databases
- (3) Change and analyze data
- (4) Get data back in reports, web pages, etc

MySQL SQL is MySQL implementation of ANSI SQL standard

MySQL SQL is close to but it is not identical to ANSI SQL standard

mysql command line interface is an enhancement of MySQL SQL

MySQL Workbench is another Graphical User Interface (GUI) to MySQL SQL

Introduction to Structured Query Laguage (SQL)

Outline

- Structured Query Language
- Characteristics
- Functionality
- Formatting

Functionality

SQL consists of:

Data definition statements:

- `CREATE TABLE`,
- `CREATE INDEX`,
- `CREATE VIEW`,
- `ALTER TABLE`,
- ...

Data retrieval statements:

- `SELECT`
- `WITH`
- ...

Functionality

Data manipulation statements:

- `UPDATE`,
- `INSERT`,
- `DELETE`,
- ...

Access control statements:

- `GRANT`,
- `REVOKE`,

System administration statements:

- `CREATE DATABASE`,
- `CREATE TABLESPACE`,
- `ALTER TABLESPACE`,
- ...

Introduction to Structured Query Laguage (SQL)

Outline

- Structured Query Language
- Characteristics
- Functionality
- Formatting

Formatting

SQL is NOT case sensitive as long as case sensitivity is set up in a different way in a particular system, e.g. MySQL

SELECT statement

```
SELECT EMPLOYEE.*, DEPARTMENT.*
FROM EMPLOYEE, DEPARTMENT
WHERE EMPLOYEE.dname = DEPARTMENT.dname;
```

SELECT statement

```
SELECT EMPLOYEE.*, DEPARTMENT.*
from EMPLOYEE, DEPARTMENT
WHERE EMPLOYEE.DNAME = DEPARTMENT.dname;
```

SELECT statement

```
select EMPLOYEE.*, DEPARTMENT.*
FROM EMPLOYEE, DEPARTMENT
WHERE EMPLOYEE.dname = DEPARTMENT.DNAME;
```

SELECT statement

```
select EMPLOYEE.*, DEPARTMENT.*
from EMPLOYEE, DEPARTMENT
WHERE EMPLOYEE.DNAME = DEPARTMENT.DNAME;
```

Formatting

The literal values in MySQL SQL statements are case sensitive

Literals in SELECT statement

```sql
SELECT CONCAT('Number: ', enum ), CONCAT('Full name :', ENAME)
FROM EMPLOYEE;
```

Literals in SELECT statement

```sql
SELECT CONCAT('NUMBER: ', enum ), CONCAT('FULL NAME :', ENAME)
FROM EMPLOYEE;
```

Literals in SELECT statement

```sql
SELECT CONCAT('Number: ', enum ), CONCAT('Full name :', ENAME)
FROM EMPLOYEE
WHERE DNAME = 'Sales';
```

Literals in SELECT statement

```sql
SELECT CONCAT('Number: ', enum ), CONCAT('Full name :', ENAME)
FROM EMPLOYEE
WHERE DNAME = 'SALES';
```

Formatting

SQL statements are terminated with a semicolon

When a statement is terminated with a semicolon, then it is immediately processed by a database server

When a statement is not terminated with a semicolon then the command line interface opens a new line for continuation of the statement.

Single line SELECT statement

```
SELECT  ENUM "Employee number", ENAME "Full name" FROM EMPLOYEE;
```

Multiline SELECT statement

```
SELECT ENUM "Employee number",
-> ENAME "Full name"
-> FROM EMPLOYEE;
```

Formatting

SQL statements can be formatted in any way as long as keywords operations and literals can be properly recognized by a compiler

Correctly formatted SELECT statement

```
SELECT ENUM "Employee number", ENAME "Full name"
FROM EMPLOYEE;
```

Correctly formatted SELECT statement

```
SELECT ENUM "Number",
       ENAME "Full name"
FROM EMPLOYEE;
```

Correctly formatted SELECT statement

```
SELECT ENUM "Employee number", ENAME "Full name" FROM
EMPLOYEE;
```

- Formatting below is incorrect

Incorrectly formatted SELECT statement

```
SELECT ENUM"Employee number",ENAME"Full name"FROMEMPLOYEE;
```

References

C. Coronel, S. Morris, A. Basta, M. Zgola, Data Management and Security, Chapter 5 Cengage Compose eBook, 2018, eBook: Data Management and Security, 1st Edition

T. Connoly, C. Begg, Database Systems, A Practical Approach to Design, Implementation, and Management, Chapter 6.1 Introduction to SQL, Pearson Education Ltd, 2015

D. Darmawikarta, SQL for MySQL A Beginner's Tutorial, Introduction, Brainy Software Inc. First Edition: June 2014