CPS221 - SOFTWARE SYSTEMS

AN INTRODUCTION TO THE SYNTAX OF SQL

General form of the SELECT statement

The following is the general form of the SQL `SELECT` statement, including most of the available options. Any clause enclosed in square brackets is optional. (The square brackets themselves do not appear in the `SELECT` statement, of course.) Italics are used to describe values that must be supplied for a given clause.

```
SELECT [ DISTINCT ] select expression(s)
      FROM table(s)
      [ WHERE selection condition ]
      [ GROUP BY column(s) ]
      [ HAVING selection condition ]
      [ ORDER BY column(s) ]
```

The meaning of the various parts of the statement is as follows:

`DISTINCT`	Ordinarily, if processing a query would result in two or more identical rows being printed, all rows would be printed. DISTINCT causes only one copy of each row to be printed.
	
	Example: If we used the query
	
	`SELECT DEPARTMENT FROM CURRENT_TERM_COURSES`
	
	each department name would appear in the result once for each course that department is teaching. If we used
	
	`SELECT DISTINCT DEPARTMENT FROM CURRENT_TERM_COURSES`
	
	each department would be printed just once

select expression(s)	Column names, functions, or expressions, separated by commas, or * to print all columns.
`FROM` *table(s)*	Either a single table, or two or more joined tables.
selection condition (`WHERE, HAVING`)	A boolean expression involving comparisons - can be a compound expression constructed using AND, OR, NOT. The selection condition in the WHERE clause (if present) is applied before groups are formed by any GROUP BY clause; the selection condition in the HAVING clause (if present) is applied after groups are formed and should only be used in queries using GROUP BY
`GROUP BY`	If this clause is present, rows are grouped by the specified column(s). In this case, the selection expressions should only be columns used for forming the groups and/or aggretgate functions such as SUM(), COUNT(), MAX() etc.

Some Further Examples of the SELECT Statement, Showing the Use of Various Features of this Statement

1.Print a report showing the names of all faculty teaching a course that meets at 8 AM. Each faculty member's name should appear at most once, and the names should be listed in alphabetical order.

```
SELECT DISTINCT PROFESSOR_NAME
  FROM TEACHES NATURAL JOIN CURRENT_TERM_COURSES
  WHERE START_TIME = '0800'
  ORDER BY PROFESSOR_NAME;
```

2.Print a report showing students majoring in CS and number of credits each is taking - but only include students taking at least 8 credits. Print in alphabetical order of name.

```
SELECT LAST_NAME, FIRST_NAME, SUM(CREDITS)
  FROM STUDENTS NATURAL JOIN ENROLLED_IN NATURAL JOIN COURSES_OFFERED
  WHERE MAJOR = 'CS'
  GROUP BY LAST_NAME, FIRST_NAME
  HAVING SUM(CREDITS) >= 8
  ORDER BY LAST_NAME, FIRST_NAME;
```

Syntax for SQL Statements to Modify the Database

1.Insert new row(s) (most commonly used forms):

```
INSERT INTO table [ (columns) ]
  VALUES  (expressions)

INSERT INTO table [ (columns) ]
  SELECT ...
```

The meaning of the various parts of the statement is as follows:

columns Names of columns into which data is to be inserted. If the list of columns is omitted, data will be inserted into all columns of the table. If columns are specified, any columns not named will be given their default value (typically NULL); if this is precluded by a NOT NULL declaration on the column then the statement will be rejected.

VALUES The values to be inserted into the row are explictily listed in the statement - either constants, or expressions involving constants. A column set by an earlier entry can be used in a later one. One new row is inserted.

expressions Values to be inserted into the various columns. If the list of columns is omitted, there must be one expression for each column declared for the table, and expressions are matched with columns by order of declaration. If columns are specified, then there must be one expression for each column listed, and the first expression is inserted into the first column listed, the second expression into the second column ...
SELECT The result of the select statement (which must have the appropriate number of columns as described for VALUES above) is used as the data to insert. As many new rows are inserted as there are rows in the result of the SELECT.

2.Delete row(s):

```
DELETE FROM table
  [ WHERE condition ]
```

The meaning of the various parts of the statement is as follows:

condition Those rows meeting the condition are deleted. If WHERE and a condition are omitted, all rows are deleted!

3.Modify existing row(s):

```
UPDATE table
  SET column = expression [, column = expression ] ...
  WHERE condition
```

The meaning of the various parts of the statement is as follows:

column = expression Each column is set to the specified value, which may be an expression computed based on the previous value of the column

condition Those rows meeting the condition are updated. If WHERE and a condition are omitted, all rows are updated.

Examples of Using the Above Statements to Modify the Database

Assume a table called `EMPLOYEES` with columns `SSN, LAST, FIRST, DEPT,` and `SALARY`

1.Add a new employee - all values given

```
INSERT INTO EMPLOYEES
  VALUES ('123-45-6789', 'AARDVARK', 'ANTHONY', 'CS', 1000000.00);
```

2.Add a new employee - salary and department not determined yet - note columns can be specified in any desired order

```
INSERT INTO EMPLOYEES(FIRST, LAST, SSN)
  VALUES ('ANTHONY, 'AARDVARK', '123-45-6789');
```

3.Delete all employees with salary greater than 100000 (maybe we're into cost-saving!)

```
DELETE FROM EMPLOYEES
  WHERE SALARY > 100000;
```

4.Give all employees in the 'CS' department a 20% pay increase

```
UPDATE EMPLOYEES
  SET SALARY = SALARY * 1.20
  WHERE DEPT = 'CS';
```