SQL - Manipulating tables

Data Definition Language (DDL)

The SQL language has facilities to create, manipulate and delete (drop) tables. Often these command line activities are duplicated through a GUI (such as the one in phpMyAdmin), however there are advantages to performing these operations through text. As an example consider a temporary table created, filled with records and then dropped with no user intervention.

Creating a table

The SQL create table syntax is of the form

```
CREATE TABLE tablename
     (column_name    type [NULL/NOT NULL],
      column_name    type [NULL/NOT NULL],
          ..)
```

According to the relational model rules, each tablename must be unique for the database and each *column_name* must be unique for each relation/table.

column_name may be up to 30 characters in length starting with an initial alphabetic character. The name may consist of alphanumeric characters and the special characters _ , $, # or @ .

The SQL standard suggests the following types:

CHAR (size)	Character data, maximum of 'size' characters upto 255
DATE	Dates (which include time)
TEXT	Character data up to 65535 (some restrictions may apply on the use of this field in a select statement)
INT	Maximum of 40 digits (will accept scientific notation)

There are other types which can be found in the SQL standard or the user guide for the particular database you are using.

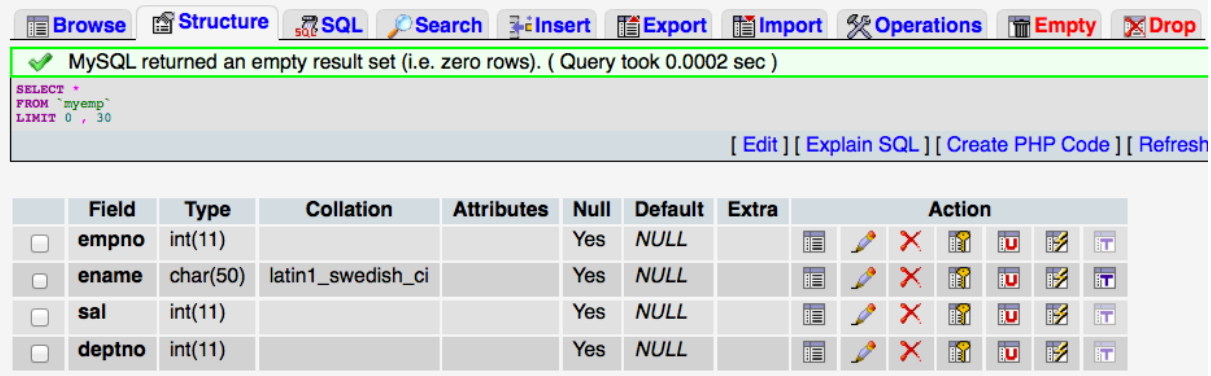
NULL and NOT NULL indicate whether the field will allow NULL values (which is the default) or whether all cells must have a value.

Activity: Try the following DDL statement out

The following DDL code will create a table called *myemp* with four fields.

```
CREATE TABLE myemp (
  empno int(11) NOT NULL,
  ename char(50),
  sal int(11),
  deptno int(11)
)
```

This *myemp* table has four fields: employee no (empno), employee name (ename), salary (sal), department no (deptno). NULL values will not be allowed in the *empno* field.



Additional information about a field (such as whether it's a primary or foreign key, or automatically incremented) can also be appended to a field description. It is common for a primary key value to be auto incremented so that newly created records will have a valid key field value (+1 from the previous higher value).

Modifying a table structure

To change a table structure use the *alter table* command:

In SQL:

```
alter table tablename
     ( [MODIFY columnname type |
        ADD columnname type ] )
```

Activity: Try the following DDL statements out

To add a *spouses_name* field to the *myemp* table

```
alter table myemp add spouses_name char(50)
```

To increase the size of the *ename* column in SQL:

```
alter table myemp modify ename char(60)
```

To remove a column in SQL:

```
alter table myemp drop spouses_name
```

Note that different databases will react in different ways if attempts are made to

- Delete columns where there is data present
- Decrease the size of a column where data is present
- Change NULL columns to NOT NULL
- Convert a column to a different type

Some implementations will take unpredictable 'best guess' solutions.

Data manipulation Language (DML)

Most SQL queries allow views on the original data, without manipulating the original data set. Actual changes to rows (records or tuples) in a table are done through the Insert, Update or Delete statements.

Inserting records into a table

The insert statement adds records (rows) to a table and has two forms:

```
insert into table [(columnname, columnname, …)]   values
(value, value,…)
```

This will insert a record using a supplied column list the supplied values. If no column list is supplied the record will be inserted as is, which may generate errors if the columns don't match up.

```
insert into table [(columnname, columnname, …)]   select
select-list
     from table(s) … etc
```

This form allows an insert to be based on the results of a *select* query.

Activity: Try the following DML statements out

Inserting a new record into the *dept* table:

```
insert into myemp (empno, ename, sal, deptno)
     values (12, ”Brown”, 35000, 3)
```

If a field is left off the list but is defined as NOT NULL an error message will be generated. Note that autoincrementing key fields can be left off the insert list and the appropriate values will be calculated and pasted in.

Inserting staff members from dept 2 into a table called *myemp*:

```
insert into myemp (empno, ename, sal, deptno)
select empno, ename, sal, deptno from emp where deptno=2;
```

Deleting records in a table

The delete statement removes records (rows) from a table:

```
delete from table [where condition]
```

Note that if no condition is supplied, all records may be deleted (leaving an empty table structure).

Activity: Try the following DML statements out

Deleting two rows, which match the empno supplied

```
DELETE
FROM myemp
WHERE empno=405 or deptno=824
```

Deleting all the rows

```
DELETE
FROM myemp
```

Deleting a table

To permanently delete a table (to 'drop' a table), use the *drop* command:

```
drop table tablename
```

Note that most databases regard this as a irreversible process – no undo features are typically supplied

---

Activity: Try the following DDL statement out

Deleting the *myemp* table:

```
drop table myemp
```

Activities: Writing your own DDL and DML statements

Write SQL commands to perform the following activities

1: Create a specialised employee table called *empLondon*, which has the *empno*, *ename*, *job*, *sal* and *deptno* columns

2: Write an appropriate SQL query to insert a new member of staff into *empLondon* with the following details:

Employee no: 697 Ename: Parsons
Job: Manager Sal: 27000 Dept: 1

3: Write a query that will insert the details of other personnel (who work in London) into the table

4: Write a query to increase the salary of everyone in the empLondon table by 15%.

5:Write an SQL statement to drop the empLondon table

6: (Harder) mySQL doesn't support the minus key – see if you can implement a version using selection, projection and cartesian croduct that will solve the following - Write a query that will show the salaries for all staff who are not earning the highest salary for their grade.