Basic DDL and DML Statement

Overview

SQL statements are divided into two major categories: **data definition language (DDL)** and **data manipulation language (DML)**. Recall Database Management system > Data Definition Language (DDL) and Database Management system > Data Manipulation Language (DML).

Data Definition Language

DDL statements are used to build and modify the structure of you tables and other objects in the database. When you execute a DDL statement, it takes effect immediately.
Create table statement:

```
CREATE TABLE <table name>(
<attribute name 1> <data type 1>,
...
<attribue name n><data type n>);
```

Data Types

- For character and Strings, `VARCHAR` or `CHAR` for variable or fixed length strings.
- For numeric types, `NUMBER` or `INTEGER`, which will usually specify a precision.
- etc.

Alter Table

The alter table statement may be used as you have seen to specify primary and foreign key constraints, as well to make other modification to the table structure. Key constraints may also be specified in the `CREATE TABLE` statement.

```
ALTER TABLE <table name>
ADD CONSTRAINT <constraint name>
PRIMARY KEY(<attribute list>);
```

If you totally mess things up and want to start over, you can always get rid of any object you've created with a drop statement.

```
DROP TABLE <table name>;

ALTER TABLE <table name>
DROP CONSTRAINT <constraint name>;
```

Data dictionary is where all the information about objects in you schema is contained.

Data Manipulation Language

DML statement are used to work data in tables.
Select statement is used in retrieving data rather than modifying it.

The insert table is used to add new rows to a table.

```
INSERT INTO <table name>
VALUES (<value 1>,...<value n>);
```

The update statement is used to change values that are already in a table

```
UPDATE <table name>
SET <attribute> = <expression>
WHERE <condition>;
```

The delete statement works for rows in a table.

```
DELETE FROM <table name>
WHERE <condition>;
```

If you want to save

```
COMMIT;
```

If you messed things up

```
ROLLBACK;
```

Note

A single-user system don't support **commit** and **rollback** statements, they are used in large systems to control **transaction**, which are sequences of changes to the database.

Privilege

If you want anyone else to be able to view or manipulate the data in your tables, and if your system permits this, you will have to explicitly **grant** the appropriate privilege to them.

```
GRANT select, insert ON customers TO webuser;
```

SELECT Statements

An SQL `SELECT` statement retrieves record from a database table according to clauses that specify criteria.

```
SELECT column1,comlumn2 FROM table1, table2 WHERE column2='value';
```

Explanation

- The `SELECT` clause specifies one or more columns to be retrieved, use coma and space between column name for multiple column. To retrieve all columns, use `*`
- The `FROM` clause specifies one or more table to be queried.
- The `WHERE` clause selects only the rows in which the specified column contains the specified value. The value is enclosed in single quotes.
- The semicolon is the statement terminator.

Example

To select all column from a table (`Customer`) for rows where the `Last_name` column has `Smith` for its value, you would send this `SELECT` statement to the server back end:

```
SELECT * FROM Customer WHERE Last_name='Smith';
```

The server back end would reply with a result set similar to this:

Cust_no	Last_name	First_name
1001	Smith	John
2039	Smith	David
2098	Smith	Matthew

To return only the `Cust_no` and `First_name` columns, based on the same criteria as above, use this statement:

```
SELECT Cust_no, First_name FROM Customers WHERE Last_name='Smith';
```

The subsequent result set might look like:

Cust_no	First_name
1001	John
2039	David
2098	Matthew

To make `WHERE` clause find inexact matches, add the pattern-matching operator `LIKE`. The `LIKE` operator uses the `%` to match zero or more characters, and the `_` to match exactly one character.

To select the `First_name` and `Nickname` columns from the `Friends` table for rows is which the `Nickname` column contains the string "brain", use this statement:

```
SELECT First_name, Nickname FROM Friends WHERE Nickname LIKE '%brain';
```

The subsequent result set might look like:

First_name	Nickname
Ben	Brainiac
Glen	Peabrain
Steven	Nobrainer

To query the same table, retrieving all columns for rows in which the `First_name` column's value begins with any letter and ends with "en", use this statement:

```
SELECT * FROM Friends WHERE First_name Like '_en';
```

The result set might look like:

First_name	Last_name	Nickname
Ben	Smith	Brainiac
Jen	Peters	Sweetpea

If you used the `%` (`'%en'`) instead in the example above, the result set might look like:

First_name	Last_name	Nickname
Ben	Smith	Brainiac
Glen	Jones	Peabrain
Jen	Peters	Sweetpea
Steven	Griffin	Nobrainer

---

see also

Database Management system