ST. XAVIER’S COLLEGE

(Affiliated to Tribhuvan University)

Maitighar, Kathmandu



Database Management System

Lab Assignment #8

Submitted by:

Biraj Poudyal  
013BSCCSIT015

Submitted to:

|  |  |
| --- | --- |
| Er. Sanjay Kumar Yadav  Lecturer  St. Xavier’s College |  |

Date of Submission: Sunday, September 27, 2015

**4.2 Data Definition language.**

A data definition language (DDL) is a computer language used to create and modify the structure of database objects in a database. These database objects include views, schemas, tables, indexes, etc.

Commonly used DDL in SQL querying are:

* CREATE: This command builds a new table and has a predefined syntax. The CREATE statement syntax is CREATE TABLE [table name] ([column definitions]) [table parameters]. CREATE TABLE Employee (Employee Id INTEGER PRIMARY KEY, First name CHAR (50) NULL, Last name CHAR (75) NOT NULL).
* ALTER: An alter command modifies an existing database table. This command can add up additional column, drop existing columns and even change the data type of columns involved in a database table. An alter command syntax is ALTER object type object name parameters. ALTER TABLE Employee ADD DOB Date.
* DROP: A drop command deletes a table, index or view. Drop statement syntax is DROP object type object name. DROP TABLE Employee.

**4.2.1. Domain Type in SQL**

char(n): Fixed length character string, with user-specified length n.

varchar(n): Variable length character strings, with user-specified maximum length n.

int: Integer (a finite subset of the integers that is machinedependent).

smallint: Small integer (a machine-dependent subset of the integer domain type).

numeric(p,d): Fixed point number, with user-specified precision of p digits, with n digits to the right of decimal point.

real, double precision: Floating point and double-precision floating point numbers, with machine-dependent precision.

float(n): Floating point number, with user-specified precision of at least n digits.

**4.2.2. Scheme Definition in SQL.**

A schema is a collection of database objects (as far as this hour is concerned—tables) associated with one particular database username. This username is called the schema owner, or the owner of the related group of objects. You may have one or multiple schemas in a database. Basically, any user who creates an object has just created his or her own schema. So, based on a user's privileges within the database, the user has control over objects that are created, manipulated, and deleted. A schema can consist of a single table and has no limits to the number of objects that it may contain, unless restricted by a specific database implementation.

**4.3. Data Manipulation Language.**

SQL is equipped with data manipulation language (DML). DML modifies the database instance by inserting, updating and deleting its data. DML is responsible for all froms data modification in a database. SQL contains the following set of commands in its DML section −

* SELECT/FROM/WHERE
* INSERT INTO/VALUES
* UPDATE/SET/WHERE
* DELETE FROM/WHERE

These basic constructs allow database programmers and users to enter data and information into the database and retrieve efficiently using a number of filter options.

**4.3.1 The Select Clause**

The SELECT statement is used to select data from a database.

The result is stored in a result table, called the result-set.

Sql select syntax

SELECT *column\_name*,*column\_name*  
FROM *table\_name*;

and

SELECT \* FROM *table\_name*;

**4.3.2 The Where Clause**

The SQL WHERE clause is used to specify a condition while fetching the data from single table or joining with multiple tables. If the given condition is satisfied then only it returns specific value from the table. You would use WHERE clause to filter the records and fetching only necessary records. The WHERE clause is not only used in SELECT statement, but it is also used in UPDATE, DELETE statement, etc.

**Syntax**

The basic syntax of SELECT statement with WHERE clause is as follows:

**SELECT column1, column2, column N FROM table\_name**

**WHERE [condition]**

**Example**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Name | Address | Phone | Salary |
| 1 | Ram | Kathmandu | 553344 | 5000 |
| 2 | Shyam | Pokhara | 667788 | 3000 |
| 3 | Hari | Bhaktapur | 445566 | 7000 |
| 4 | Bir | Chitwan | 223344 | 2000 |

SQL Select query:  **SELECT id, name FROM Info WHERE salary > 4000**

The query above is processed and following result appears:

|  |  |
| --- | --- |
| Id | Name |
| 1 | Ram |
| 3 | Hari |

**4.3.3 The Form Clause**

The sql FROM clause is used to select table while retrieving data from a table . The statement FROM is not only used for retrieving data but also used for any join operation.

**Syntax**

The basic syntax of SQL FROM is:

**SELECT column1,column2,…..columnN FROM table\_number**

**Example:**

SQL query: SELECT \* FROM Info

**4.3.4 The Rename Operation**

With Rename operation, we can rename a table and give a new name.

**Syntax:**

**RENAME TABLE table\_name TO new\_table\_name**

**Example**

SQL query: RENAME TABLE info To detail\_info

**4.3.5 Tuple Variable**

1. Tuple variables can be used in SQL, and are defined in the **from** clause:

**select distinct** *cname, T.loan#*

**from** *borrower* ***as*** *S, loan* ***as*** *T*

**where** *S.loan# = T.loan#*

Note: The keyword **as** is optional here.

1. These variables can then be used throughout the expression. Think of it as being something like the rename operator.

Finds the names of all branches that have assets greater than at least one branch located in Burnaby.

**select distinct** *T.bname*

**from** *branch S, branch T*

**where** *S.bcity=``Burnaby''* **and** *T.assets > S.assets*

**4.3.6 String Operations**

1. The most commonly used operation on strings is pattern matching using the operator **like**.
2. String matching operators **%** (any substring) and **\_** (underscore, matching any character).

E.g., ``\_\_\_%'' matches any string with at least 3 characters.

1. Patterns are case sensitive, e.g., ``Jim" does not match ``jim".
2. Use the keyword **escape** to define the *escape* character.

E.g., like ``ab%tely tex2html_wrap_inline1742 % tex2html_wrap_inline1742 '' escape `` tex2html_wrap_inline1742 '' matches all the strings beginning with ``ab'' followed by a sequence of characters and then ``tely'' and then ``% tex2html_wrap_inline1742 ''.

Backslash overrides the special meaning of these symbols.

1. We can use **not like** for string mismatching.
2. Example. Find all customers whose street includes the substring ``Main''.

**select** *cname*

**from** *customer*

**where** *street* **like** *``%Main%''*

1. SQL also permits a variety of functions on character strings, such as concatenating (using `` tex2html_wrap_inline1750 ''), extracting substrings, finding the length of strings, converting between upper case and lower case, and so on.

**4.3.7 Ordering The Display Of Tuples.**

**4.3.8 Duplicate Tuples**

|  |  |
| --- | --- |
|  |  |