Introduction to Structured Query Language (SQL)

Introduction to SQL

- Data Definition Language (DDL): SQL includes commands to create database objects such as tables, indexes, and views, as well as commands to define access rights to those database objects.
- Data Manipulation Language (DML): SQL includes commands to insert, update, delete, and retrieve data within the database tables.
- Data Control Language (DCL): used to control access to data stored in a database.

Data Definition Commands

Creating the Database

- Creates the physical files that will hold the database.
- Interacts with the operating system and the file systems supported by the operating system.
- Syntax:

  CREATE DATABASE <*database name*>;

Connecting to a Database

- Based on IBM DB2
- Syntax: **connect to** <*database name*> [**user** <*username*> **using** <*password*>]

The Database Schema

- Schema is a group of database objects that are related to each other.
- Schema belongs to a single user or application.
- enforces first level of security by allowing each user to see only the tables that belong to that user.
- ANSI SQL Syntax:

  CREATE SCHEMA AUTHORIZATION <*creator name*>;
- DB2 Syntax:

  CREATE SCHEMA <schema name> | **AUTHORIZATION** <authorization name> |<schema name> **AUTHORIZATION** <authorization name>

Creating Table Structures

- **Syntax**:

  CREATE TABLE <[schema name.]*table name*> (
  column1 data type [constraint] [,
  column2 data type [constraint]] [,
  PRIMARY KEY *(column1 [, column2])*] [,
  FOREIGN KEY *(column1 [, column2*]) REFERENCES *tablename*] [,
  CONSTRAINT *constraint*]);

SQL Constraints

- PRIMARY KEY
  - **PRIMARY KEY (column1 [,column2])**
    - *PRIMARY KEY (itemno)*
- NOT NULL
  - **<column name> NOT NULL**
    - *LastName varchar(20) NOT NULL*

- FOREIGN KEY
  - **FOREIGN KEY (column name) REFERENCES <table name> ON [<update/delete> <action taken>]**
  - ***On update no action***
    - When a primary key of any row in the parent table is updated and results to any row in the dependent (an associative table for example) table without a corresponding parent key, the update is rejected
  - ***On update restrict***:
    - Any row whose primary key in the parent table exist as foreign key in the dependent table, its update is rejected

EMPLOYEE	
EmpId	EmpLastname
1000	Dela Cruz
1001	Mustaine
1002	Ulrich

DEPENDENT		
DepId	DepLastname	EmpId
D1000	Dela Cruz	1000
D1001	Dela Cruz	1000
D1002	Mustaine	1001

- ***On delete cascade***:
  - If any row in the parent table is deleted, then all rows in the dependent table whose foreign key matches the primary key of the deleted row are also deleted.
- ***On delete restrict (or No action)***:
  - If any row in the parent table is deleted and its value exists as a foreign key value in the dependent table, the delete command will be rejected.
- ***On delete set NULL***:
  - Each nullable column of the foreign key of each dependent is set to null.

- UNIQUE
  - Makes sure that no duplicate exists; used with NOT NULL
  - **<column name> <data type> UNIQUE**
  - **CONSTRAINT <index name> UNIQUE (column name, …)**
  - **UNIQUE (column name, …)**
    - *Custlname varchar(20) NOT NULL UNIQUE*
    - *CONSTRAINT pid UNIQUE (id, custlastname)*
    - *UNIQUE (custlastname,custfirstname)*

- DEFAULT
  - assigns a value to an attribute when a new row is added to a table
  - **<column name> <data type> DEFAULT <default value>**
    - *Area_code char(4) DEFAULT '6200'*

- CHECK
  - used to validate data when an attribute value is entered
  - **<column name> <data type> CHECK(<condition>)**
  - **CONSTRAINT <check name> CHECK(<condition>)**
    - *Area_code char(4) CHECK (area_code in ('6200','6230'))*
    - *Area_code char(4) DEFAULT '6200' CHECK (area_code in ('6200','6230'))*
    - *CONSTRAINT code_ck1 CHECK (area_code in ('6200','6230'))*

Indexes

- Are special lookup tables that the database search engine can use to speed up data retrieval. An orderly arrangement used to logically access rows in a table
- Composed of an index key and a set of pointers.
- **CREATE [UNIQUE] INDEX indexname ON tablename(column1 [, column2])**
  - *CREATE UNIQUE itemidx on items(itemid)*
- **DROP INDEX indexname**
  - *DROP INDEX itemidx*

Data Manipulation Commands

Adding Table Rows

- **INSERT INTO <tablename>[(*column*$_1$, *column*$_2$, …, *column*$_n$)] VALUES (*value*$_1$, *value*$_2$, … , *value*$_n$)**
  - *Insert into items values(10,'pencil',500,15.25)*
  - *Insert into items values(10,'pencil',500,NULL)*
  - *Insert into items(itemno, itemName, itemqty) values (10,'pencil',500)*
  - *Insert into items(itemName, itemqty, price) values ('pencil',500,15.25)*

Select Statement

- **SELECT <column list> FROM <table list> [WHERE <condition list>];**
  - *SELECT itemno, itemName, itemprice, itemqty FROM items;*
  - *SELECT itemno, itemName, itemprice, itemqty FROM items where itemno = 1001;*
  - *SELECT itemno, itemName, itemprice, itemqty FROM items where itemName = 'pencil';*
  - *SELECT * FROM items where itemName = 'pencil';*
  - SELECT itemno, itemName, itemprice, itemqty, itemqty * itemprice FROM items;
  - SELECT itemno, itemName, itemprice, itemqty, itemqty * itemprice as totalamount FROM items;

SQL Operators

Logical Operators

- **AND, OR, AND NOT**
  - *SELECT * FROM items WHERE itemno = 1001 or itemno = 1010;*
  - *SELECT * FROM items WHERE itemqty >= 100 and itemqty <= 500;*
  - *SELECT * FROM items WHERE not(itemno = 1001);*
  - *SELECT * FROM items WHERE itemno <>1001*

Special Operators

- **BETWEEN** – checks whether an attribute value is within the range.
  - *SELECT * FROM items WHERE itemprice between 10 and 20.50;*
- **IS NULL** – checks whether an attribute value is null.
  - *SELECT FROM items WHERE itemprice IS NULL;*

- **LIKE** – checks whether an attribute value matches a given string pattern.
  - used in conjunction with wildcards to find patterns within string attributes.
  - Uses the percent sign (%) and underscore (_) wildcard characters.
    - *SELECT * FROM items WHERE itemName like 'p%';*
    - *SELECT * FROM items WHERE itemName like '%n';*
    - *SELECT * FROM items WHERE itemName like '_ _ _ paper';*
    - *SELECT * FROM items WHERE itemName like '%p%';*

- **IN** Operator
  - Can be used in place of the OR operator
    - *SELECT * FROM items WHERE itemno = 1001 OR itemno = 1006 OR itemno = 1100*
    - *SELECT * FROM items WHERE itemno IN (1001, 1002, 1100)*

Updating Table Rows

- **UPDATE <table name> SET <column name> = <expression> [, <column name> = <expression>] [WHERE <condition list>];**
  - *UPDATE items SET itemqty = 40;*
  - *UPDATE items SET itemqty = 40 WHERE itemno = 1001;*
  - *UPDATE items SET itemtotal = itemqty * itemprice WHERE itemno = 1001;*

Deleting Table Rows

- **DELETE FROM <table name> [WHERE <condition list>];**
  - *DELETE FROM items;*
  - *DELETE FROM items WHERE itemno = 1100;*

Inserting Table Rows with Select Subquery

- **INSERT INTO <table name> SELECT <column list> FROM <table name>**;
  - INSERT INTO items_2 SELECT * FROM items;

Advanced Data Definition Commands

Alter Table

- Used to change a table's structure
- **Adding a new column**
  - Syntax: **ALTER TABLE** <*table name*> **ADD COLUMN** <*column name*> <*data type*> [column constraint]
  - Examples:
    - Alter table table1 add column address varchar(50)
    - Alter table table1 add column address varchar(50) not null default 'Dumaguete City'

- *Alter table table1 add column address varchar(50) not null default 'Dumaguete City' add column balance decimal(7,2) default 0.0*

- **Removing a column**
  - Syntax: **ALTER TABLE** <table name> **DROP COLUMN** <column name>
  - Examples:
    - *alter table product drop column subtotal*
    - *Alter table product drop column quantity drop column subtotal*

- **Modifying the length & data type**
  - Syntax: **ALTER TABLE** <table name> **ALTER COLUMN** <column name> <set data type new type/new length>
  - Examples:
    - alter table orders alter column created_date set data type timestamp
    - alter table orders alter column note set data type varchar(255);

- **Modifying Default Values**
  - **Syntax: ALTER TABLE** <table name> **ALTER COLUMN** <column name> **SET DEFAULT** <new default value>
  - Example:
    - Alter table table1 alter column balance set default 100

- **Renaming a column**
  - Syntax: **ALTER TABLE** <table name> **RENAME COLUMN** <column name> **TO** <new column name>
  - Example:
    - Alter table table1 rename column name to studname

Advanced Select Queries

Sorting a List

- **SELECT <column list> FROM <table list> [WHERE <condition list>] [ORDER BY <column list> [ASC | DESC]] ;**
  - *SELECT * FROM items ORDER BY itemqty DESC;*
  - *SELECT * FROM items ORDER BY itemprice, itemqty;*

Grouping Data

- **SELECT <column list> FROM <table list> [WHERE <condition list>] [GROUP BY <column list>] [HAVING <condition list>] [ORDER BY <column list>[ASC | DESC]]** ;
  - *SELECT dept, name FROM staff GROUP BY dept, name ORDER BY dept;*

End of Presentation

Sources:

Rob, Peter and Coronel, Carlos. *Database Systems : Design, Implementation and Management*, 7th Edition. Course Technology, Thomson Learning Inc. ©2007

https://www.ibm.com/support/producthub/db2/docs/content/SSEPGG_11.5.0/com.ibm.db2.luw.admin.dbobj.doc/doc/c0020153.html