Intro To Database

(Database Fundamental using SQLSERVER)

Mohamed ELshafei

SQL

Structured Query Language

SQL

Intake 39 - DB

SQL

- SQL (pronounced "ess-que-el") stands for **Structured Query Language**.
- SQL is a database computer language designed for the retrieval and management of data in a relational database
- Developed in the early 1974 (SEQual)
- ANSI-SQL defined by the American National Standards Institute
- SQL is a language to operate databases; it includes database creation, deletion, fetching rows, modifying rows, etc.

SQLSERVER Features

- High Performance.
- High Availability.
- Robust Transactional Support.
- Strong Data Protection.

Categories of MySQL Statements

- **DML – Data Manipulation Language** :refers to the INSERT, UPDATE and DELETE statements , DML allows to add / modify / delete data itself.
- **DCL – Data Control Language** : refers to the GRANT and REVOKE statements
- **DDL – Data Definition Language** : refers to the CREATE, ALTER and DROP statements , DDL allows to add / modify / delete the logical structures
- **DTL - Data Transaction Language** :refers to the START TRANSACTION, SAVEPOINT, COMMIT and ROLLBACK [TO SAVEPOINT] statements
- **DQL - Data Query Language** (Select) :refers to the SELECT, SHOW and HELP statements (queries)

Data Types

A data type determines the type of data that can be stored in a database column. The most commonly used data types are:

1. Alphanumeric: data types used to store characters, numbers, special characters, or nearly any combination.
2. Numeric
3. Date and Time
4. other data types

Database Constraints

- Not Null.
- Primary Key.
- Unique Key.
- Referential Integrity (FK).
- CHECK
- DEFAULT

Create Command

Create table "table_name" ("column name" data type, "column name" data type, ...)

Example (1)

```
CREATE TABLE customer
(ID int (3) Not Null, First_Name char(50), Last_Name char(50),
 City char(25), Birth_Date date, Primary key (ID)
FOREIGN KEY (PersonID) REFERENCES Persons(PersonID));
```

Example (2)

```
CREATE TABLE customer
(ID int (15) Primary key, First_Name char(50), Last_Name

 char(50),  City char(25), Birth_Date date);
```

Constraints

```
CREATE TABLE Student
(
ID int(6) NOT NULL,
NAME varchar(10) NOT NULL,AGE int NOT NULL CHECK (AGE >= 18)
);
```

```
CREATE TABLE Student
(
ID int(6) NOT NULL,
NAME varchar(10) NOT NULL,
AGE int DEFAULT 18
);
```

Constraints

```
CREATE TABLE Orders
(ID int NOT NULL,
ORDERNO int NOT NULL,CID int,
PRIMARY KEY (ID),
FOREIGN KEY (CID)
REFERENCES Customers(CID)
)
```

```
CREATE TABLE new_employees
(
 idnum int IDENTITY(1,1),
 fname varchar (20),
 minit char(1),
 lname varchar(30)
);
```

Drop command

```
Drop table “table name”;
```

✓ Drop table Customer

Alter command

```
ALTER TABLE table_name ADD column_name datatype
```

```
ALTER TABLE table_name DROP COLUMN column_name
```

Example:

- ✓ ALTER TABLE Customer ADD Address char(40)
- ✓ ALTER TABLE Customer DROP COLUMN Address

DML -Data Manipulation Language

- Insert.
- Update.
- Delete.

INSERT Command

Person table

LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo

✓ INSERT INTO "table_name” VALUES ("value1", "value2", ...)

- **Insert a New Row:**

  INSERT INTO Person VALUES (‘Saleh’, ‘Ahmed', ‘Moharam bak', ‘Alex.')

Person table

LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo
Saleh	Ahmed	Moharam bak.	Alex.

INSERT Command (cont.)

Person table

LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo

- **Insert a New Row:**

INSERT INTO Person (LastName, City) VALUES (‘Hassan', ‘Assuit')

Person table

LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo
Hassan			Assuit.

Update Command

✓ UPDATE "table_name"
SET "column_1" = {new value}
[WHERE {condition}]

Example (1)

```
UPDATE Person
SET City= ‘Assiut’
```

All records will be updated

Example (2)

```
UPDATE Person
SET City= ‘Assiut’

Where FirstName = ‘Ahmed’
```

Only records with first name ‘Ahmed’ will be updated

Update Command (cont.)

✓ Update several Columns in a Row:

LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo
Saleh	Ahmed	Moharam bak.	Alex.

```
UPDATE Person
SET       Address = ‘241 El-haram ',  City = ‘Giza'
WHERE   LastName = ‘El-Sayed'
```

LastName	FirstName	Address	City
El-Sayed	Mohamed	241 El-haram	Giza
Saleh	Ahmed	Moharam bak.	Alex.

Delete Command

✓ DELETE FROM "table_name"
[WHERE {condition}]

Example (1)

```
DELETE FROM Person
```

All records will be deleted

Example (2)

```
DELETE FROM Person
Where FirstName = ‘Ahmed’
```

Only records with first name ‘Ahmed’ will be deleted

DQL

```
Select <attribute list >
From  < table list>
[ Where <condition> ]
```

- ✓
```
select *
from department;
```

- ✓
```
select emp_id, emp_name, dept_id
from   employee;
```

- ✓
```
select distinct dept_id
from employee;
```

SELECT with Condition

```
Select  dept_id, dept_name
from    department
where   location = ‘Cairo’;
```

Comparison Conditions

- = Equal.
- > greater than.
- >= greater than or equal.
- < less than.
- <= less than or equal.
- <>not equal.

```
Select   last_name, salary
from     employee
where    salary >1000
```

Logical Conditions

- AND.

```
Select   last_name, salary
from     employee
where  city = ‘Assiut’ and salary > 1000;
```

- OR.

```
Select   last_name, salary
from     employee
where  city = ‘Assiut’ OR salary > 1000;
```

- NOT.

```
Select   emp_id, last_name, salary, manager_id
From     employee
where   manager_id NOT IN (100, 101, 200);
```

Other Comparison Conditions

- Between …… AND ….. (between two values - Inclusive).

```
Select   last_name, salary
from     employee
where  salary between 1000 and 3000;
```

- IN (set) (Match any of a list of values)

```
Select    emp_id, last_name, salary, manager_id
From      employee
where    manager_id IN (100, 101, 200);
```

- Like (Match a character Pattern)

```
Select   first_name
from      employee
where   first_name Like ‘s%’;
```

Arithmetic Expressions

```
Select   last_name, salary, salary + 300
from     employee;
```

- Order of precedence: * , / , +, -
- You can enforce priority by adding parentheses.

```
Select   last_name, salary, 10 * (salary + 300)
from     employee;
```

Order by Clause

- It is used to sort results either in ascending or descending order.

```
✓ Select      fname, dept_id, hire_date
  From        employee
  Order by    hire_date  [ ASC ];

✓ Select      fname, dept_id, hire_date
  From        employee
  Order by    hire_date  DESC;

✓ Select      fname, dept_id, salary
  From        employee
  Order by    dept_id, Salary  DESC;
```

THANKS!

Any questions?