Crash Course in SQL Statements

Mr. Ferddie Q. Canlas

What is SQL?

- Databases speak Structured Query Language, better known as SQL,
- Invented by E. F. Codd in the 1970s.
- Instead of working with tables one record at a time, SQL manages groups of records as a single entity, which makes it suitable for creating queries of any complexity.

What is SQL? (cont..)

- This language has been standardized, and now most database servers, and ADO itself, accept its ANSI-92 dialect.

2 Distinct Categories Of SQL Statements

- DDL (Data Definition Language) subset includes a group of statements that allow you to create database structures, such as tables, fields, indices, and so on
- DML (Data Manipulation Language) subset includes all the commands that allow you to query and modify the data in the database, add new records, or delete existing ones

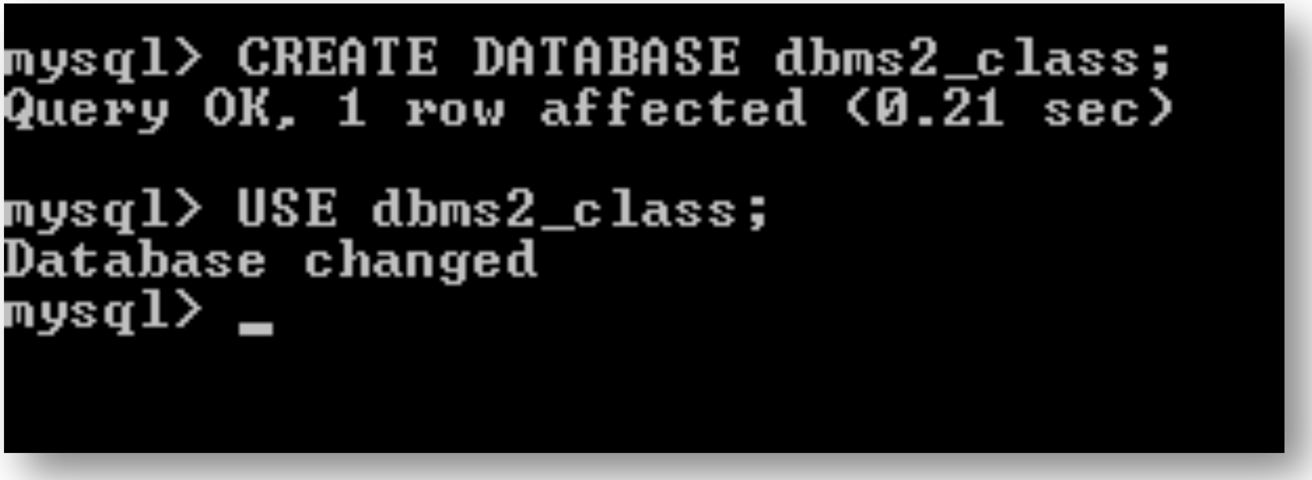
DDL Commands

Creating and Using a Database

Syntax:

CREATE DATABASE *databasename;*

USE *databasename;*



```
mysql> CREATE DATABASE dbms2_class;
Query OK, 1 row affected (0.21 sec)

mysql> USE dbms2_class;
Database changed
mysql> _
```

MySQL does not automatically make the database you just created the active database. You must implicitly state which database to activate with a USE statement.

Deleting a Database

SYNTAX:

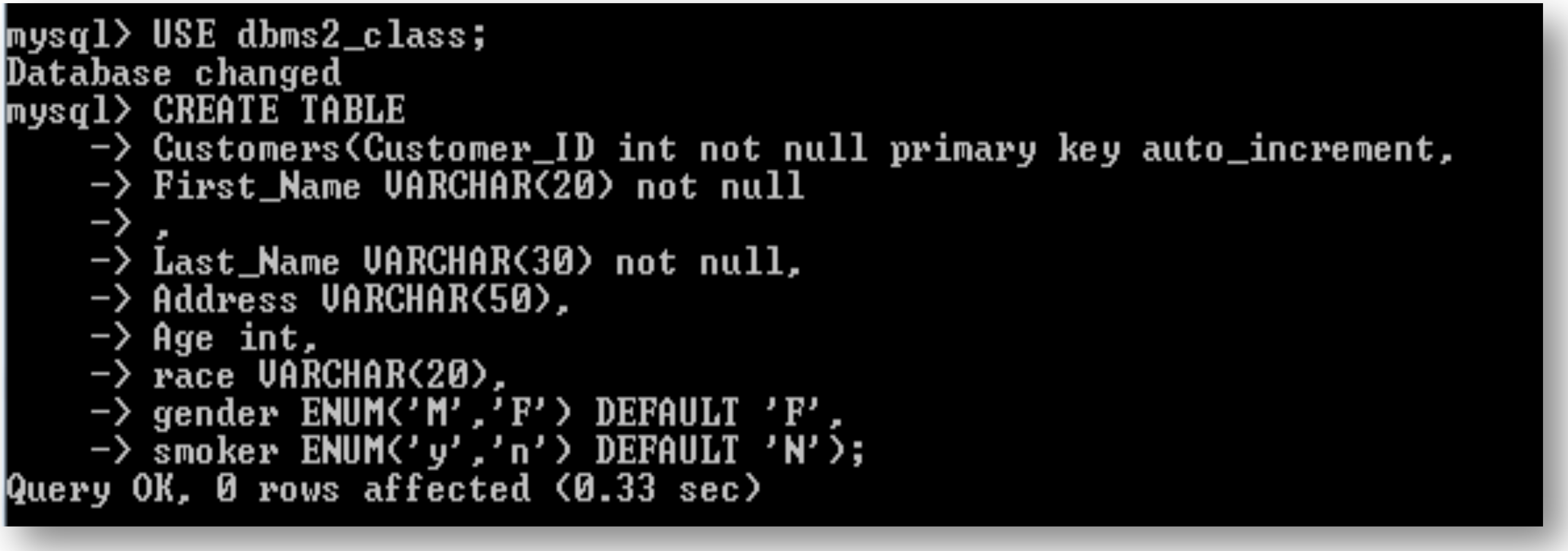
```
DROP DATABASE databasename;
```

This will delete the *databasename* and ALL the data within it.

Creating Tables

Syntax:

CREATE TABLE *tablename (columnname datatype);*



```
mysql> USE dbms2_class;
Database changed
mysql> CREATE TABLE
    -> Customers(Customer_ID int not null primary key auto_increment,
    -> First_Name VARCHAR(20) not null
    -> ,
    -> Last_Name VARCHAR(30) not null,
    -> Address VARCHAR(50),
    -> Age int,
    -> race VARCHAR(20),
    -> gender ENUM('M','F') DEFAULT 'F',
    -> smoker ENUM('y','n') DEFAULT 'N');
Query OK, 0 rows affected (0.33 sec)
```

The SELECT Command

- The most frequently used SQL statement is undoubtedly the SELECT command, which returns a set of records based on selection criteria.

Examples of Basic SELECT Statement

- The simplest SELECT command returns all the records and all the fields from a database table:

```
SELECT * FROM Publishers
```

- Specify fields to retrieve:

```
SELECT PubID, [Company Name], Address
FROM Publishers
```

SELECT with CRITERIA

- Exact Criteria:

  SELECT Name, City FROM Publishers WHERE State = 'CA‘

- Partial or Keyword

  SELECT * FROM Publishers WHERE

  State = 'CA' AND Name LIKE 'M%‘

SELECT with CRITERIA

- Specifying Range

```
SELECT * FROM Titles WHERE [Year Published]
BETWEEN 1996 AND 1998
```

Note: =,<,>,<=,>= and <> can also be used.

- The IN operator is useful when you have a list of values

```
SELECT Name, State FROM Publishers WHERE State
   IN ('CA', 'TX', 'NJ')
```

Sorting and Grouping

- The ORDER BY clause lets you affect the order in which records are retrieved.

  SELECT * FROM Publishers ORDER BY [Company Name] ASC

- You can also specify multiple sort keys by separating the keys with commas

  SELECT * FROM Publishers ORDER BY State, City DESC

Sorting and Grouping

- When the results are sorted, you can decide to take just the first records returned by the SELECT, which you do with the TOP clause:

```
SELECT TOP 5 * FROM Titles ORDER BY [Year Published] DESC
```

- You can define the number of returned records in terms of the percentage of the total number of the records that would be returned, using the TOP PERCENT clause:

```
SELECT TOP 10 PERCENT * FROM Titles ORDER BY [Year
   Published] DESC
```

The ISNULL and DISTINCT

- Testing the field with the ISNULL function, and you can filter out duplicates using the DISTINCT keyword:

```
SELECT DISTINCT City FROM Publishers WHERE
NOT ISNULL(City)
```

Join Commands

- **INNER JOIN**: Return rows when there is at least one match in both tables
- **LEFT JOIN**: Return all rows from the left table, even if there are no matches in the right table
- **RIGHT JOIN**: Return all rows from the right table, even if there are no matches in the left table

INNER JOIN

- SELECT column_name(s)
  FROM table_name1
  INNER JOIN table_name2
  ON table_name1.column_name=table_name2.column_name

LEFT JOIN

- SELECT column_name(s)
  FROM table_name1
  LEFT JOIN table_name2
  ON
  table_name1.column_name=table_name2
  .column_name

RIGHT JOIN SYNTAX

- SELECT column_name(s)
  FROM table_name1
  RIGHT JOIN table_name2
  ON
  table_name1.column_name=table_name2
  .column_name

UNION

- Appends two tables
- Tables should have the same number of fields

- Syntax:

“SELECT table1.fields FROM table1 UNION SELECT table2.fields” FROM table2”

INSERT INTO Command

- The INSERT INTO command adds a new record to a table and sets its fields in one operation. You must provide a list of field names and values:

```
INSERT INTO Authors (Author, [Year Born])
VALUES ('Frank Whale', 1960)
```

- Copy entire table’s content into another table:

```
INSERT INTO Authors SELECT * FROM NewAuthors
```

INSERT INTO - Combination

- You can combine WHERE clause to specify fields:

```
INSERT INTO Customers SELECT ContactName
AS Name, Address, City, State FROM Contacts
WHERE Successful = True
```

The UPDATE Command

- The UPDATE command modifies the values in one or more records. You often use a WHERE clause to restrict its action to the record(s) you're interested in:

```
UPDATE Authors SET [Year Born] = 1961 WHERE
Author = 'Frank Whale'

UPDATE [Order Details] INNER JOIN Orders ON
[Order Details].OrderID = Orders.OrderID SET
Discount = Discount + 0.10 WHERE CustomerID =
'LILAS'
```

The DELETE Command

- The DELETE command lets you remove one or more records from a table. You must append a WHERE clause to this command unless you want to delete all the records in the table:

```
DELETE FROM Titles WHERE [Year Published] < 1950
```

Thank you very much!