SQL

Tags

Table of Contents

1. Database and Table Operations

1.1 Creating and Using Database

Code:

```
CREATE DATABASE BusinessDB;
USE BusinessDB;
```

Comments:

- Creates a new database named BusinessDB
- Switches context to use the newly created database
- All subsequent operations will be performed in this database

1.2 Creating Tables

Code:

```
CREATE TABLE staff (
   id INT,
   name VARCHAR(50),
   age INT,
   gender VARCHAR(1),
   phone VARCHAR(10),
   city VARCHAR(15)
);
```

Table Structure:

Column Name	Data Type	Description
id	INT	Employee identifier
name	VARCHAR(50)	Full name
age	INT	Employee age
gender	VARCHAR(1)	Gender (M/F)
phone	VARCHAR(10)	Contact number
city	VARCHAR(15)	City of residence

1.3 Inserting Data

Code:

```
INSERT INTO staff (id, name, age, gender, phone, city)
VALUES
    (1, "Michael Scott", 19, "M", "4022155", "Chicago"),
    (2, "Pam Beesly", 21, "F", "4034421", "New York"),
    (3, "Jim Halpert", 20, "M", "4056221", "Chicago"),
    (4, "Angela Martin", 18, "F", "4098621", "Dallas"),
    (5, "Dwight Schrute", 22, "M", "405221", "Chicago");
```

Output:

id	name	age	gender	phone	city
1	Michael Scott	19	M	4022155	Chicago
2	Pam Beesly	21	F	4034421	New York
3	Jim Halpert	20	M	4056221	Chicago
4	Angela Martin	18	F	4098621	Dallas
5	Dwight Schrute	22	M	405221	Chicago

2. Data Manipulation

2.1 Alter Table Operations

Code:

```
-- Adding new column
ALTER TABLE staff ADD salary INT;

-- Modifying column position
ALTER TABLE staff MODIFY salary INT AFTER name;

-- Adding unique constraint
ALTER TABLE staff ADD UNIQUE(name);
```

Changes Made:

1. New salary column added
2. Salary column moved after name
3. Unique constraint added to name column

---

3. Query Operations

3.1 Basic Select Operations

Code:

```
-- With column aliases
SELECT id AS "Employee ID",
       name AS "Employee Name"
FROM staff;

-- All columns
SELECT * FROM staff;
```

Output:

Employee ID	Employee Name
1	Michael Scott
2	Pam Beesly
3	Jim Halpert
...	...

3.2 WHERE Clause Examples

Code:

```
-- Gender filter
SELECT * FROM staff WHERE gender = "M";

-- Age range filter
SELECT * FROM staff WHERE age BETWEEN 18 AND 21;
```

Output for gender filter:

id	name	age	gender	phone	city
1	Michael Scott	19	M	4022155	Chicago
3	Jim Halpert	20	M	4056221	Chicago
5	Dwight Schrute	22	M	405221	Chicago

3.3 Pattern Matching

Code:

```
-- Names starting with 'M'
SELECT * FROM staff WHERE name LIKE "M%";

-- Names containing 'a'
SELECT * FROM staff WHERE name LIKE "%a%";
```

3.4 Sorting Results

Code:

```
-- Ascending order by name
SELECT * FROM staff ORDER BY name ASC;

-- Descending order by age
SELECT * FROM staff ORDER BY age DESC;
```

4. String Functions

Code Examples:

```
-- Upper case conversion
SELECT id, UPPER(name) AS Name FROM staff;
```

```
-- String length
SELECT id, name, CHARACTER_LENGTH(name) AS Length FROM staff;

-- String concatenation
SELECT CONCAT(id, " - ", name) AS "Employee Info" FROM staff;
```

Output:

id	Name	Length
1	MICHAEL SCOTT	13
2	PAM BEESLY	10
...	...	...

5. Date and Time Functions

Code Examples:

```
-- Current date
SELECT CURRENT_DATE();

-- Date extraction
SELECT EXTRACT(MONTH FROM "2024-02-15 09:34:21");

-- Date addition
SELECT ADDDATE("2024-02-15", INTERVAL 10 DAY);
```

6. Aggregate Functions

Code Examples:

```
-- Count records
SELECT COUNT(*) AS "Total Staff" FROM staff;

-- Average age by city
SELECT city,
       AVG(age) AS "Average Age",
       COUNT(*) AS "Number of Staff"
FROM staff
GROUP BY city;
```

Output:

city	Average Age	Number of Staff
Chicago	20.3	3
New York	21.0	1
Dallas	18.0	1

---

7. Join Operations

7.1 Table Setup

Code:

```
CREATE TABLE departments (
    dept_id INT PRIMARY KEY,
    dept_name VARCHAR(50)
);

CREATE TABLE projects (
    project_id INT,
    project_name VARCHAR(30),
    dept_id INT
);

INSERT INTO departments VALUES
    (1, "Sales"),
    (2, "Marketing"),
    (3, "IT");
```

7.2 Join Examples

Code:

```
-- Inner Join
SELECT s.name, d.dept_name
FROM staff s
INNER JOIN departments d
ON s.dept_id = d.dept_id;

-- Left Join
SELECT s.name, d.dept_name
```

```
FROM staff s
LEFT JOIN departments d
ON s.dept_id = d.dept_id;
```

Output:

name	dept_name
Michael Scott	Sales
Pam Beesly	Marketing
Jim Halpert	IT
...	...