
DIGGIBYTE
CREATING VALUE WITH DATA

DIGGIBYTE
CREATING VALUE WITH DATA

# SQL and Pyspark

DIGGIBYTE
CREATING VALUE WITH DATA

# Agenda

- Comparing SQL and Pyspark Functions

DIGGIBYTE
CREATING VALUE WITH DATA

# Creating a DataFrame/Table

- SQL: Uses CREATE TABLE and INSERT statements
- PySpark: Uses spark.createDataFrame() to create a DataFrame from data

```sql
-- Create the employees table with
--columns for id, name, department, and salary
CREATE TABLE employees (
    id INT PRIMARY KEY,
    name VARCHAR(255),
    dept VARCHAR(255),
    salary INT
);
-- Insert sample data into the employees table
INSERT INTO employees (id, name, dept, salary) VALUES
    (1, 'Amit kundu     ', 'Sales', 50000),
    (2, '    Sidhart Shukla', 'Marketing', 60000),
    (3, 'Jim Smith', 'Sales', 55000),
    (4, 'Jane Doe', 'HR', 65000),
    (5, 'Mike Lee', 'Marketing', 70000);
```

```python
# Create the employees DataFrame
employees = spark.createDataFrame([
    (1, 'Amit kundu     ', 'Sales', 50000),
    (2, '    Sidhart Shukla', 'Marketing', 60000),
    (3, 'Jim Smith', 'Sales', 55000),
    (4, 'Jane Doe', 'HR', 65000),
    (5, 'Mike Lee', 'Marketing', 70000)
], ['id', 'name', 'dept', 'salary'])
```

DIGGIBYTE
CREATING VALUE WITH DATA

# Selecting All Rows and Columns

- SQL: Uses SELECT * FROM table_name

- PySpark: Uses dataframe.display()

```sql
-- Select all columns and rows from the employees table
SELECT * FROM employees;
```

```python
# Show all columns and rows from the employees DataFrame
employees.display()
```

DIGGIBYTE
CREATING VALUE WITH DATA

# Filter, Distinct Values, Grouping and Aggregating

```sql
-- Select all columns and rows from
--the employees table where the department is 'Sales'
SELECT *
FROM employees
WHERE dept = 'Sales';

-- Select distinct (unique) department values
--from the employees table
SELECT DISTINCT dept
FROM employees;

-- Select the department and the sum of salaries
--for each department, grouped by department
SELECT dept, SUM(salary) AS total_salary
FROM employees
GROUP BY dept;
```

```python
# Select all columns and rows where the department is
'Sales'
employees.filter(employees.dept == 'Sales').display()

# Select unique department values
employees.select('dept').distinct().display()

# Select the department and the sum of salaries for each
department
employees.groupBy('dept').sum('salary').display()
```

DIGGIBYTE
CREATING VALUE WITH DATA

# Multiple Conditions

```sql
SELECT id, name
FROM employees
WHERE dept IN ('Sales', 'Marketing');

SELECT id, name
FROM employees
WHERE dept = 'Sales'
AND salary > 60000;
```

```python
# Select the id and name columns for employees in the
Sales and Marketing departments
employees.filter(employees.dept.isin(['Sales',
'Marketing'])) \
        .select('id', 'name') \
        .display()

# Select the id and name columns for employees in the
Sales department with salary > 60000
sales_high_salary = employees.filter((employees.dept ==
'Sales') & (employees.salary > 60000)) \
                    .select('id', 'name')
sales_high_salary.display()
```

DIGGIBYTE
CREATING VALUE WITH DATA

# Ordering, Counting, Min, Max, and Averages

```sql
--ORDER BY salary
SELECT * FROM employees ORDER BY salary desc;

-- count
SELECT COUNT(*) FROM employees;

-- min, max
SELECT MIN(salary) as lowest_salary,
MAX(salary) as highest_salary FROM employees;

-- average
SELECT AVG(salary) as avg_salary FROM employees;
```

```python
from pyspark.sql.functions import min, max, mean, count

# Order by highest salary
result = employees.orderBy(employees["salary"].desc())
result.display()
# count
result = employees.count()
# min, max
min_salary = employees.select(min("salary")).collect()[0]
[0]
max_salary = employees.select(max("salary")).collect()[0]
[0]
# average
avg_salary = employees.select(mean("salary")).collect()[0]
[0]
```

DIGGIBYTE
CREATING VALUE WITH DATA

# Split Columns

- Split one column into multiple columns

```sql
-- To split the name column into first_name and last_name columns

SELECT
    id,
    SUBSTRING(name, 1, CHARINDEX(' ', name) - 1) AS first_name,
    SUBSTRING(name, CHARINDEX(' ', name) + 1, LEN(name)) AS last_name,
    dept,
    salary
FROM employees
```

```python
employees_df = employees_df.withColumn("first_name", split(col("name"), " ")[0]) \
                           .withColumn("last_name", split(col("name"), " ")[1])
```

DIGGIBYTE
CREATING VALUE WITH DATA

# Join & Union

- Join Multiple tables and Combine all rows using Union

```sql
--INNER JOIN
SELECT orders.order_id, orders.customer_id, customers.customer_name
FROM orders
JOIN customers ON orders.customer_id = customers.customer_id;

--UNION
SELECT customer_id FROM orders
UNION
SELECT customer_id FROM customers;
```

```python
## INNER JOIN
joined_df = orders_df.join(customers_df, orders_df.customer_id == customers_df.customer_id, "inner") \
                     .select(orders_df.order_id, orders_df.customer_id, customers_df.customer_name)
joined_df.display()

## UNION
unioned_df = orders_df.select("customer_id") \
                      .union(customers_df.select("customer_id"))
unioned_df.display()
```

DIGGIBYTE

CREATING VALUE WITH DATA

# THANK YOU !!