DATA ENGINEERING

Name: Ramireddy Preethi

Batch: Python Batch 2

DAY3:

DATA INTEGRITY: Data integrity refers to safety of data and ensures accuracy, consistency, and reliability of data in relational database.

4 TYPES OF DATA INTEGRITY:

1. Entity Integrity
2. User defined Integrity
3. Referential Integrity
4. Domain Integrity

1) Entity Integrity: Records should be uniquely identifiable, the primary rule of entity integrity is that every table must have a primary key or unique identifier that cannot contain NULL values.

Primary Key Constraints

USE: Prevent duplicate records

2) User defined Integrity: Allows users to specify their own rules and constraints for their specific purpose

Triggers and Custom Constraints: User-defined integrity is often implemented using triggers, stored procedures, or application-level validations.

3) Referential Integrity: This refers to relationship between two or more tables in a database and ensure that data is consistently maintained across those tables.

Foreign key Constraints

4) Domain Integrity: This refers to the restrictions placed on the type and range of data that can be stored in a column.

***** Data Type (integer/character/date/time/string/etc.) and Constraints (Not null/check/unique/primary key/foreign key/default)

```sql
INSERT INTO employees (first_name, last_name, dept, email, phone, hire_date, salary) VALUES
('Nancy1', 'Davolio', 'Sales Representative', 'nancy.davolio@example.com', '555-0123', '2023-01-15', 5000.00);
```

```
Messages
Msg 2627, Level 14, State 1, Line 20
Violation of UNIQUE KEY constraint 'uq_email'. Cannot insert duplicate key in object 'dbo.employees'. The duplicate key value is (nancy.davolio@example.com).
The statement has been terminated.

Completion time: 2024-10-30T11:00:39.1846440+05:30
```

```sql
-- Implementing Data Integrity

-- Add a constraint to ensure salary is always at least 3000
alter table employees add constraint check_salary check(salary>=3000)

-- Insert a salary < 3000 to test check constraint
INSERT INTO employees (first_name, last_name, dept, email, phone, hire_date, salary) VALUES
('Nancy1', 'Davolio', 'Sales Representative', 'nancy.davolio@example.com', '555-0123', '2023-01-15', 2000.00);
```

```
Messages
Msg 547, Level 16, State 0, Line 11
The INSERT statement conflicted with the CHECK constraint "check_salary". The conflict occurred in database "org", table "dbo.employees", column 'salary'.
The statement has been terminated.

Completion time: 2024-10-30T11:04:00.0216478+05:30
```

FUNCTIONS TO CUSTOMIZE RESULT SET:

Types of Functions:

- Aggregate Functions
- Scalar Functions

<u>Aggregate Functions:</u> operate on group of data and give single output.

NOTE: 1) GROUP BY clause is often used with aggregate functions to group rows that have same value.

2) HAVING clause is used to filter results after aggregate function has been applied. It is similar to WHERE clause but used with aggregate functions.

MIN: function returns the smallest value of the selected column.

MAX: function returns the largest value of the selected column.

COUNT: Returns the number of rows that match a specified criterion.

SUM: Returns the total sum of a numeric column.

AVG: Returns the average value of a numeric column.

<u>Scalar Functions:</u> These are In-Built functions, operate on single data and give single output.

1. **UCASE:** change the case of the string to upper case characters.
2. **LCASE:** change the case of the string to lower case characters.
3. **MID:** extract substrings from the table's column, which contain values of string type.

4. **LENGTH:** returns the length of the string in the column.
5. **ROUND:** used to round a numeric column to the number of decimals specified.
6. **NOW:** returns the current system date and time.
7. **FORMAT:** used to format how a column is to be displayed.

String Functions:

CONCAT: Concatenates two or more strings into one.

```
-- concat
select first_name,last_name,concat(first_name,last_name) as concatname from employees
```

100 %

Results | Messages

	first_name	last_name	concatname
1	Nancy	Davolio	NancyDavolio
2	Andrew	Fuller	AndrewFuller
3	Janet	Leverling	JanetLeverling
4	Margaret	Peacock	MargaretPeacock
5	Steven	Buchanan	StevenBuchanan
6	Michael	Suyama	MichaelSuyama
7	Robert	King	RobertKing
8	Laura	Callahan	LauraCallahan
9	Nancy	Smith	NancySmith
10	Andrew	NULL	Andrew
11	Michael	Johnson	MichaelJohnson
12	Laura	NULL	Laura
13	Robert	Brown	RobertBrown
14	Margaret	Thompson	MargaretThompson

SUBSTRING or SUBSTR: Extracts a substring from a string, starting at a specified position and for a specified length.

```sql
-- substring
select first_name ,SUBSTRING(first_name,2,3) as substring from employees
```

100 %

Results | Messages

	first_name	substring
1	Nancy	anc
2	Andrew	ndr
3	Janet	ane
4	Margaret	arg
5	Steven	tev
6	Michael	ich
7	Robert	obe
8	Laura	aur
9	Nancy	anc
10	Andrew	ndr
11	Michael	ich
12	Laura	aur
13	Robert	obe
14	Margaret	arg

LENGTH or LEN : Returns the length of a string (number of characters).

```sql
-- length
select first_name , len(first_name) as length from employees
```

100 %

Results | Messages

	first_name	length
1	Nancy	5
2	Andrew	6
3	Janet	5
4	Margaret	8
5	Steven	6
6	Michael	7
7	Robert	6
8	Laura	5
9	Nancy	5
10	Andrew	6
11	Michael	7
12	Laura	5
13	Robert	6
14	Margaret	8

UPPER: Converts a string to uppercase.

```sql
-- uppercase
select first_name,upper(first_name) as uppercase from employees
```

100 %

Results | Messages

	first_name	uppercase
1	Nancy	NANCY
2	Andrew	ANDREW
3	Janet	JANET
4	Margaret	MARGARET
5	Steven	STEVEN
6	Michael	MICHAEL
7	Robert	ROBERT
8	Laura	LAURA
9	Nancy	NANCY
10	Andrew	ANDREW
11	Michael	MICHAEL
12	Laura	LAURA
13	Robert	ROBERT
14	Margaret	MARGARET

LOWER: Converts a string to lowercase.

```sql
-- lowercase
select first_name,lower(first_name) as lowercase from employees
```

100 %

Results | Messages

	first_name	lowercase
1	Nancy	nancy
2	Andrew	andrew
3	Janet	janet
4	Margaret	margaret
5	Steven	steven
6	Michael	michael
7	Robert	robert
8	Laura	laura
9	Nancy	nancy
10	Andrew	andrew
11	Michael	michael
12	Laura	laura
13	Robert	robert
14	Margaret	margaret

TRIM: Removes leading and trailing spaces from a string.

REPLACE: Replaces all occurrences of a specified substring with another substring.

```sql
-- Update email domains from example.com to company.com
SELECT first_name, last_name, email, REPLACE(email, '@example.com', '@company.com') AS updated_email FROM employees;
```

100 %

Results | Messages

	first_name	last_name	email	updated_email
1	Nancy	Davolio	nancy.davolio@example.com	nancy.davolio@company.com
2	Andrew	Fuller	andrew.fuller@example.com	andrew.fuller@company.com
3	Janet	Leverling	janet.leverling@example.com	janet.leverling@company.com
4	Margaret	Peacock	margaret.peacock@example.com	margaret.peacock@company.com
5	Steven	Buchanan	steven.buchanan@example.com	steven.buchanan@company.com
6	Michael	Suyama	michael.suyama@example.com	michael.suyama@company.com
7	Robert	King	robert.king@example.com	robert.king@company.com
8	Laura	Callahan	laura.callahan@example.com	laura.callahan@company.com
9	Nancy	Smith	nancy.smith@example.com	nancy.smith@company.com
10	Andrew	NULL	andrew@example.com	andrew@company.com
11	Michael	Johnson	michael.johnson@example.com	michael.johnson@company.com
12	Laura	NULL	laura@gmail.com	laura@gmail.com
13	Robert	Brown	robert.brown@example.com	robert.brown@company.com
14	Margaret	Thompson	Margaret@example.com	Margaret@company.com

LEFT: Returns a specified number of characters from the left side of a string.

RIGHT: Returns a specified number of characters from the right side of a string.

CHARINDEX (SQL Server) / INSTR (MySQL, Oracle): Returns the position of a specified substring within a string.

```sql
-- Using String Functions
-- Find employees with their email domains
SELECT first_name, last_name, email, CHARINDEX('@',email) AS domain FROM employees;

-- Update email domains from example.com to company.com
SELECT first_name, last_name, email, REPLACE(email, '@example.com', '@company.com') AS updated_email FROM employees;
```

100 %

Results | Messages

	first_name	last_name	email	domain
1	Nancy	Davolio	nancy.davolio@example.com	14
2	Andrew	Fuller	andrew.fuller@example.com	14
3	Janet	Leverling	janet.leverling@example.com	16
4	Margaret	Peacock	margaret.peacock@example.com	17
5	Steven	Buchanan	steven.buchanan@example.com	16
6	Michael	Suyama	michael.suyama@example.com	15
7	Robert	King	robert.king@example.com	12
8	Laura	Callahan	laura.callahan@example.com	15
9	Nancy	Smith	nancy.smith@example.com	12
10	Andrew	NULL	andrew@example.com	7
11	Michael	Johnson	michael.johnson@example.com	16
12	Laura	NULL	laura@gmail.com	6
13	Robert	Brown	robert.brown@example.com	13
14	Margaret	Thompson	Margaret@example.com	9

FORMAT: Formats a string based on a specified format.

```sql
-- Format the hire date in a readable format
SELECT first_name, last_name, hire_date, FORMAT(hire_date, '%M ,%d, %y') AS formatted_hire_date FROM employees;
SELECT first_name, last_name, hire_date, FORMAT(hire_date, '%y-%M-%d') AS formatted_hire_date FROM employees;
SELECT first_name, last_name, hire_date, FORMAT(hire_date, 'yyyy-MM-dd') AS formatted_hire_date FROM employees;
SELECT first_name, last_name, hire_date, FORMAT(GETDATE(), 'yyyy-MM-dd') AS formatted_hire_date FROM employees;
```

100 %

Results | Messages

	first_name	last_name	hire_date	formatted_hire_date
1	Nancy	Davolio	2023-01-15	1 ,15, 23
2	Andrew	Fuller	2022-03-10	3 ,10, 22
3	Janet	Leverling	2023-05-20	5 ,20, 23
4	Margaret	Peacock	2023-02-25	2 ,25, 23
5	Steven	Buchanan	2022-12-01	12 ,1, 22
6	Michael	Suyama	2023-04-11	4 ,11, 23
7	Robert	King	2023-07-05	7 ,5, 23
8	Laura	Callahan	2023-08-15	8 ,15, 23

	first_name	last_name	hire_date	formatted_hire_date
1	Nancy	Davolio	2023-01-15	23-1-15
2	Andrew	Fuller	2022-03-10	22-3-10
3	Janet	Leverling	2023-05-20	23-5-20
4	Margaret	Peacock	2023-02-25	23-2-25
5	Steven	Buchanan	2022-12-01	22-12-1
6	Michael	Suyama	2023-04-11	23-4-11
7	Robert	King	2023-07-05	23-7-5
8	Laura	Callahan	2023-08-15	23-8-15

	first_name	last_name	hire_date	formatted_hire_date
1	Nancy	Davolio	2023-01-15	2023-01-15
2	Andrew	Fuller	2022-03-10	2022-03-10
3	Janet	Leverling	2023-05-20	2023-05-20
4	Margaret	Peacock	2023-02-25	2023-02-25
5	Steven	Buchanan	2022-12-01	2022-12-01
6	Michael	Suyama	2023-04-11	2023-04-11
7	Robert	King	2023-07-05	2023-07-05
8	Laura	Callahan	2023-08-15	2023-08-15

	first_name	last_name	hire_date	formatted_hire_date
1	Nancy	Davolio	2023-01-15	2024-10-30
2	Andrew	Fuller	2022-03-10	2024-10-30

String Functions:

1) **CURDATE (MYSQL)/ GETDATE (SQL SERVER):** Returns the current date.

2) **DATEPART:** Returns a specific part of a date, such as year, month, day, etc.

```sql
-- DATEPART
select DATEPART(YEAR,GETDATE()) AS CurrentYear
select DATEPART(MONTH,GETDATE()) AS CurrentMonth
select DATEPART(DAY,GETDATE()) AS CurrentDay
```

100 %

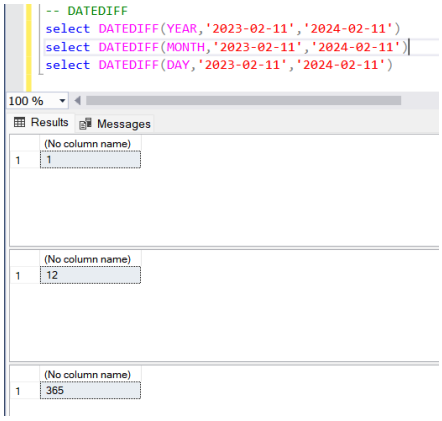
Results | Messages

	CurrentYear
1	2024

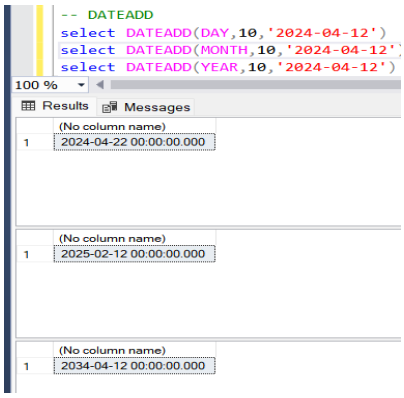
	CurrentMonth
1	10

	CurrentDay
1	30

3) DATEDIFF: Calculates the difference between two dates.



4) DATEADD: Adds a specified number of time intervals (days, months, years) to a date.



5) EXTRACT: Retrieves subparts of a date (like year, month, day).

MATHEMATICAL FUNCTIONS:

1) ABS: Give Absolute value that is positive value

2) CEILING: Rounds a number down to the nearest integer.

3) ROUND: Rounds a number to a specified number of decimal places.

4) POWER: Returns the value of a number raised to a specified power.

5) SQRT: Returns the square root of a number.

6) EXP: Returns the value of *e* raised to the power of a specified number (exponential).

7) RAND: Returns a random number between `0` and `1`. You can pass a seed to generate a repeatable sequence.