**SQL Functions Used**

1. **GETDATE()**
   * **Purpose**: Returns the current system date and time.
   * **Example Usage**:

sql

CopyEdit

WHERE CONVERT(DATE, GETDATE()) = [SomeDateColumn]

1. **CONVERT()**
   * **Purpose**: Converts a value from one data type to another.
   * **Example Usage**:

sql

CopyEdit

CONVERT(DATE, GETDATE())

* + **Used To**: Ensure date comparison is done correctly by stripping time.

1. **DATEPART()**
   * **Purpose**: Extracts a specific part of a date (e.g., year, month, day).
   * **Example Usage**:

sql

CopyEdit

DATEPART(YEAR, JoiningDate)

1. **YEAR()**
   * **Purpose**: Returns the year part of a date.
   * **Example Usage**:

sql

CopyEdit

YEAR(BirthDate)

1. **MONTH()**
   * **Purpose**: Returns the month part of a date.
   * **Example Usage**:

sql

CopyEdit

MONTH(JoiningDate)

1. **DAY()**
   * **Purpose**: Returns the day of the month from a date.
   * **Example Usage**:

sql

CopyEdit

DAY(HireDate)

1. **DATEDIFF()**
   * **Purpose**: Returns the difference between two dates in the specified unit (e.g., day, month, year).
   * **Example Usage**:

sql

CopyEdit

DATEDIFF(DAY, StartDate, EndDate)

1. **ISNULL()**
   * **Purpose**: Replaces NULL with a specified replacement value.
   * **Example Usage**:

sql

CopyEdit

ISNULL(Salary, 0)

1. **CAST()**
   * **Purpose**: Converts an expression from one data type to another.
   * **Example Usage**:

sql

CopyEdit

CAST(Salary AS INT)

Let me know if you’d like these categorized further (e.g., Date/Time functions, Conversion functions), or explained with examples from your document.

4o

**You said:**

based on the attached file provide ur inputs

**ChatGPT said:**

Based on the attached file **“SQL Commands for Task 3.docx”**, here are the **functions and key SQL features used**, categorized for clarity:

**🔧 Database & Table Creation**

* **CREATE DATABASE**
  + Creates a new database (TASK3SQLDATAANALYSIS).
* **USE**
  + Selects the created database for subsequent operations.
* **CREATE TABLE**
  + Defines two tables: orders\_data and returns\_data with specific data types.

**📥 Data Insertion**

* **INSERT INTO**
  + Inserts bulk data into orders\_data and returns\_data.

**🔍 Basic Data Retrieval**

* **SELECT**
  + Retrieves all or specific columns.
  + Examples:
    - SELECT \* FROM orders\_data;
    - SELECT order\_id, order\_date, sales FROM orders\_data;
* **SELECT TOP (n)**
  + Retrieves the top n records.
  + Example: SELECT TOP 10 \* FROM orders\_data;

**📌 Filtering Data**

* **WHERE**
  + Filters rows based on condition.
  + Examples:
    - WHERE city = 'Henderson'
    - WHERE profit > 100
* **AND / OR**
  + Combines multiple conditions.
  + Examples:
    - WHERE category = 'Furniture' AND profit > 100
    - WHERE city = 'Los Angeles' OR order\_date > '2020-11-08'

**📊 Sorting Data**

* **ORDER BY**
  + Sorts results by one or more columns.
  + Examples:
    - ORDER BY order\_date
    - ORDER BY order\_date DESC, profit DESC

**📐 Aggregate Functions**

* **SUM()**
  + Calculates total.
  + Example: SUM(quantity) AS Sum\_of\_Sales
* **MAX()**
  + Retrieves the maximum value.
  + Example: SELECT MAX(SALES) AS TOTAL\_SALES

**📦 Grouping Data**

* **GROUP BY**
  + Aggregates data by category.
  + Example:

sql

CopyEdit

SELECT category, SUM(sales), SUM(profit)

FROM orders\_data

GROUP BY category;

* **HAVING**
  + Filters results from GROUP BY.
  + Example:

sql

CopyEdit

SELECT city, SUM(sales) AS city\_sales

FROM orders\_data

WHERE region = 'West'

GROUP BY city

HAVING SUM(sales) > 500;

**🔗 Joins**

* **INNER JOIN**
  + Combines rows with matching order\_id.
* **LEFT JOIN, RIGHT JOIN**
  + Includes unmatched rows from one table.

**🔍 Indexing**

* **CREATE INDEX**
  + Speeds up retrieval on returns\_data(order\_id).

**🪟 Views**

* **CREATE VIEW**
  + Defines a virtual table (City) showing customers in Henderson.

**✅ Summary of Functions Used**

| **Function / Keyword** | **Description** |
| --- | --- |
| CREATE DATABASE | Creates a new database |
| USE | Selects database context |
| CREATE TABLE | Defines table schema |
| INSERT INTO | Adds data records |
| SELECT, TOP | Retrieves data |
| WHERE, AND, OR | Filters records |
| ORDER BY | Sorts records |
| SUM(), MAX() | Aggregate computations |
| GROUP BY, HAVING | Groups data and filters aggregated rows |
| JOIN, LEFT/RIGHT JOIN | Combines related data across tables |
| CREATE INDEX | Optimizes search queries |
| CREATE VIEW | Creates a virtual table |