DENSE_RANK() :

Window function in SQL that assigns a rank to rows within a partition, **without skipping ranks** when there are duplicates.

Syntax

sql Copy Edit

```sql
DENSE_RANK() OVER (
    PARTITION BY column_name
    ORDER BY column_name ASC|DESC
)
```

- `PARTITION BY` (Optional) – Divides the result into groups.
- `ORDER BY` – Defines the ranking order.
- **Ranks are continuous** – No gaps if values repeat.

EX:

```sql
SELECT
    name, department, salary,
    DENSE_RANK() OVER (PARTITION BY department ORDER BY salary DESC) AS dept_rank
FROM Employees;
```

Output

Name	Department	Salary	Dept_Rank
Alice	HR	7000	1
John	HR	5000	2
Bob	IT	8000	1
Charlie	IT	8000	1
David	IT	2	2

Explanation

- Employees **within each department** are ranked **separately**.
- No gaps in ranking **within each department**.

Without the “partition by” the function gives the rank based on all data irrespective of the department.

Rank():

Window function that assigns a rank to each row within a partition, but **skips ranks** when there

are duplicate values.

Syntax

sql Copy Edit

```sql
RANK() OVER (
    PARTITION BY column_name
    ORDER BY column_name ASC|DESC
)
```

- `PARTITION BY` (Optional) – Divides the data into groups (each group is ranked separately).
- `ORDER BY` – Defines ranking order (ascending or descending).
- **Ranks can have gaps** – If two or more rows have the same value, they receive the **same rank**, and the next rank is **skipped**.

EX:

Copy Edit

```sql
SELECT
    name, marks,
    RANK() OVER (ORDER BY marks DESC) AS rank_num
FROM Students;
```

Output

Name	Marks	Rank_Num
Alice	95	1
Charlie	95	1
Bob	90	3
Eve	90	3
David	85	5

Explanation

- **Alice & Charlie** have **same marks (95)** → Both get **Rank 1**.
- **Bob & Eve** have **same marks (90)** → Both get **Rank 3** (**Rank 2 is skipped**).
- **David** is next → Gets **Rank 5** (**Rank 4 is skipped**).

⚠ **Notice the ranking gaps!**

Extract():

retrieves specific date or time parts (like year, month, day, hour, etc.) from a **date, timestamp, or interval** value.

Syntax

```sql
EXTRACT(field FROM source)
```

- `field` → The part of the date/time you want to extract (e.g., `YEAR`, `MONTH`, `DAY`, `HOUR`).
- `source` → A `DATE`, `TIMESTAMP`, or `INTERVAL` column/value.

Supported Fields in Oracle

Field	Description
`YEAR`	Extracts the year (e.g., `2024`)
`MONTH`	Extracts the month (1-12)
`DAY`	Extracts the day of the month (1-31)
`HOUR`	Extracts the hour (0-23) **(only for TIMESTAMP)**
`MINUTE`	Extracts the minute (0-59) **(only for TIMESTAMP)**
`SECOND`	Extracts the second (0-59) **(only for TIMESTAMP)**
`TIMEZONE_HOUR`	Extracts the time zone hour offset
`TIMEZONE_MINUTE`	Extracts the time zone minute offset
`TIMEZONE_REGION`	Extracts the time zone region name
`TIMEZONE_ABBR`	Extracts the time zone abbreviation

❌ Fields NOT Supported in Oracle

Some fields that work in PostgreSQL **do not work in Oracle**:

Field	Description
`WEEK` ❌	Week of the year (**Use** `TO_CHAR(date, 'IW')` **instead**)
`DOW` ❌	Day of the week (**Use** `TO_CHAR(date, 'D')`)
`DOY` ❌	Day of the year (**Use** `TO_CHAR(date, 'DDD')`)
`QUARTER` ❌	Quarter of the year (**Use** `TO_CHAR(date, 'Q')`)

3 Extracting Quarter (Alternative)

Oracle does **not support** `QUARTER` in `EXTRACT()`, so use `TO_CHAR()` instead:

```sql
SELECT TO_CHAR(DATE '2024-03-22', 'Q') AS Quarter FROM dual;
```

Output

Quarter
1

4 Extracting Week (Alternative)

```sql
SELECT TO_CHAR(DATE '2024-03-22', 'IW') AS WeekNumber FROM dual;
```

Output

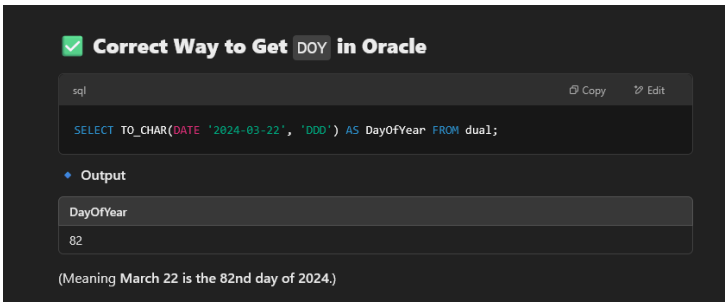
WeekNumber
12

5 Extracting Day of the Week (Alternative)

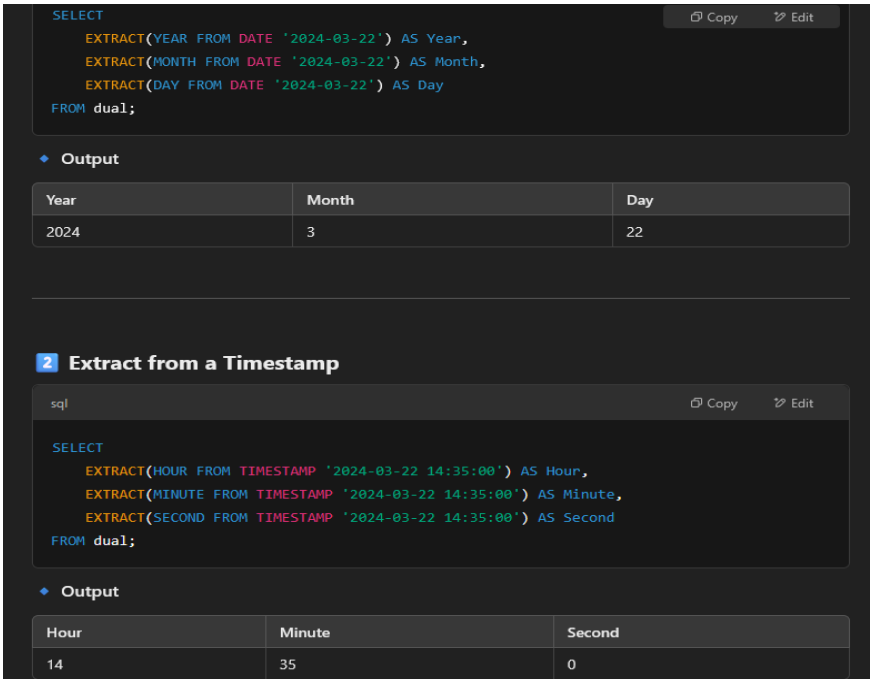
```sql
SELECT TO_CHAR(DATE '2024-03-22', 'D') AS DayOfWeek FROM dual;
```

Output

DayOfWeek
6


✅ Correct Way to Get DOY in Oracle
sql
Copy
Edit
SELECT TO_CHAR(DATE '2024-03-22', 'DDD') AS DayOfYear FROM dual;
🔹 Output
DayOfYear
82
(Meaning March 22 is the 82nd day of 2024.)

EX:


SELECT
EXTRACT(YEAR FROM DATE '2024-03-22') AS Year,
EXTRACT(MONTH FROM DATE '2024-03-22') AS Month,
EXTRACT(DAY FROM DATE '2024-03-22') AS Day
FROM dual;
Copy
Edit
🔹 Output
Year
Month
Day
2024
3
22
2️⃣ Extract from a Timestamp
sql
Copy
Edit
SELECT
EXTRACT(HOUR FROM TIMESTAMP '2024-03-22 14:35:00') AS Hour,
EXTRACT(MINUTE FROM TIMESTAMP '2024-03-22 14:35:00') AS Minute,
EXTRACT(SECOND FROM TIMESTAMP '2024-03-22 14:35:00') AS Second
FROM dual;
🔹 Output
Hour
Minute
Second
14
35
0