2024/05/21

# Unions

Combine rows together

UNION ALL: show all data.

UNION DESTINCT: remove duplicate.

CASE:

-- the department want to find out the old employees and also highly paid employees to push them out or reduce their salary to reduce the company budgets

# String Function

**LENGHTN(item):** length of the item

**UPPER( item) & LOWER( item )**: turn the letters into upper case or lower case

**TRIM( item) & LTRIM( item) &RTRIM( item):** get rid of all white spaces, left, right white space

**LEFT ( item, Nr.characters you wanna select) & RIGHT( item, , Nr.characters you wanna select)**: how many characters do you select from left right,

**SUBSTRING(item, start position, Nr.characters you wanna select**): how many characters do you select from any position you want to select, e.g. from birth\_date select the birth\_month

**REPLACE(item, characters before replacement, characters after replacement )**: replace specific characters

**LOCATE(character, item)**: identify the position of the character in the item.

**CONCAT(item,item,item…)**: concatenation of multiple columns. Eg. First\_name+Last\_name = full\_name

# Case Statement

CASE

WHEN….THEN…

WHEN….THEN…

END AS ….

放在SELECT里面用

# Subqueries

2024/05/26

窗口函数（Window Function）是在 SQL 中用于执行计算操作的函数，这些计算在特定的数据行集（称为窗口）内进行。与聚合函数不同的是，窗口函数不会将多行合并为单一输出行，而是保留原有的行数，并在其基础上添加计算结果。窗口函数常用于高级数据分析，例如计算排名、运行总计、移动平均等。

SELECT column1,

column2,

window\_function() OVER (PARTITION BY column1 ORDER BY column2)

FROM table\_name;

窗口函数的定义包含几个部分：

* **窗口函数名**：如 **ROW\_NUMBER()**、**RANK()**、**SUM()** 等。
* **OVER 子句**：定义窗口的范围和顺序。
  + **PARTITION BY**：定义窗口分区的列（类似于 **GROUP BY**，但窗口函数不合并行）。
  + **ORDER BY**：定义窗口内的行顺序。

窗口函数提供了一种强大的方式来进行复杂的数据分析和计算，不会减少结果集的行数。常见的窗口函数包括 **ROW\_NUMBER()**、**RANK()**、**DENSE\_RANK()** 和聚合函数（如 **SUM()**、**AVG()**）等，它们在分析排名、累计计算、运行总计等方面非常有用。

# CTEs

-- CTEs

-- Common Table expression, allow you to define a subquery block

-- that you can reference in main query

-- you can only use CTE you needed it after you create it

-- purpose 1: make more advanced calculations which is not easy do in one query

-- puepose 2: readability

# Temporary Table

* Only visible to the session that they are created in.
* Application:
  + Useful to restore the intermediate results for complex queries.
  + Manipulate Data before I insert into a more permanent table
* How to create

1st way

**CREATE TEMPORARY TABLE** temp\_table

( first\_name varchar(50),

last\_name varchar(50),

favorite\_movie varchar(100)

);

SELECT \*

FROM temp\_table;

INSERT INTO temp\_table

VALUES('Alex', 'Freberg', 'Lord of the rings: the two towers');

2nd way

* Note: when you create it, even you create a new window, it also works.

However, if you exit out the MySQL and come back, the temporary table is no longer to be working

# Stored Procedure

DELIMITER $$

CREATE PROCEDURE name(**parameter**)

BEGIN

…

END $$

DELIMITER ; -- change back the delimiter

# Triggers and events

Trigger: a block of code that executes automatically when an event takes place on a specific table

图形用户界面, 文本, 应用程序, 日程表

描述已自动生成