在做数据分析时，我们会经常听到同比、环比的概念。各个企业和组织在发布统计数据时，通常喜欢用同比、环比来和之前的历史数据进行比较，用来说明数据的变化情况。例如，统计局公布2022年1月份CPI同比增长0.9%，环比增长0.6%。

实际在基于数据库的数据分析场景中，环比和同比是典型的复杂计算场景之一，特别是在Oracle等商业数据库的分析函数出现之前。以MySQL为例，在8.0版本中才引入了Lag和Lead函数，这两个函数结合开窗函数有效提高了同比、环比等复杂运算的实现效率。在5.x系列版本中，MySQL需要依赖多次嵌套子查询和自关联才能实现此类计算。

我们以一个简单的例子，来分别看下，MySql 5.x和8.0是具体实现同比、环比计算的。

示例数据见表：

CREATETABLEsales (`产品ID`varchar(20), `销售数量`int(20) , `销售时间`timestamp(6) NULLDEFAULTNULL) INSERTINTOsales VALUES('C1001', 15, '2020-06-01 10:10:12'); INSERTINTOsales VALUES('C1002',26, '2020-05-02 0:10:12'); INSERTINTOsales VALUES('C1003', 21, '2020-04-03 0:10:12'); INSERTINTOsales VALUES('C1003', 23, '2020-04-04 0:10:12'); INSERTINTOsales VALUES('C1003', 0, '2020-03-05 0:10:12'); INSERTINTOsales VALUES('C1001', 16, '2020-02-06 3:0:12'); INSERTINTOsales VALUES('C1002', 32, '2020-01-07 0:10:12'); INSERTINTOsales VALUES('C1001', 16, '2019-12-08 0:12:24'); INSERTINTOsales VALUES('C1001', 32, '2019-06-09 0:12:24'); INSERTINTOsales VALUES('C1002', 17, '2019-05-09 0:12:24');

产品ID	销售数量	销售时间
C1001	15	2020-06-01 10:10:12.0000
C1002	26	2020-05-02 00:10:12.0000
C1003	21	2020-04-03 00:10:12.0000
C1003	23	2020-04-04 00:10:12.0000
C1003	0	2020-03-05 00:10:12.0000
C1001	16	2020-02-06 03:00:12.0000
C1002	32	2020-01-07 00:10:12.0000
C1001	16	2019-12-08 00:12:24.0000
C1001	32	2019-06-09 00:12:24.0000
C1002	17	2019-05-09 00:12:24.0000

1、MySQL 5.x：通过子查询和关联实现同比和占比计算

以按年月统计不同年份的销售总值，并计算环比（销售总额同比上期）、同比（销售总额同比去年同期）为例

示例表结构和数据

通过SQL计算环比和同比：

selectyear(c.销售时间) yy,month(c.销售时间) mm,
concat(ifnull(abs(round((sum(c.销售数量)-ss1)/ss1*100,2)),0),'%') 同比,
concat(ifnull(abs(round((sum(c.销售数量)-ss2)/ss2*100,2)),0),'%') 环比
fromsales c leftjoin(selectmonth(a.销售时间) mm1, year(a.销售时间) yy1,
sum(a.销售数量) ss1 fromsales a GROUPBYmm1,yy1) a onmonth(c.销售时间)
= a.mm1 anda.yy1 = year(c.销售时间)-1leftjoin(selectmonth(a.销售时间)
mm2, year(a.销售时间) yy2, sum(a.销售数量) ss2 fromsales a
GROUPBYmm2,yy2) b on(b.yy2 = year(c.销售时间) andb.mm2+1=
month(c.销售时间) OR(yy2=year(c.销售时间)-1ANDb.mm2 =
12ANDmonth(c.销售时间) = 1)) groupbyyy, mm orderbyyy,mm asc

计算结果：

yy	mm	同比	环比
2019	5	0.00%	0.00%
2019	6	0.00%	88.24%
2019	12	0.00%	0.00%
2020	1	0.00%	100.00%
2020	2	0.00%	50.00%
2020	3	0.00%	100.00%
2020	4	0.00%	0.00%
2020	5	152.94%	2.27%
2020	6	46.88%	9.30%
2020	12	0.00%	0.00%
2021	1	0.00%	100.00%
2021	2	0.00%	50.00%
2021	3	0.00%	100.00%
2021	4	0.00%	0.00%
2021	5	39.53%	40.91%
2021	6	68.09%	42.31%

2、MySQL 8.0：通过分析函数实现同比和占比计算

MySql8.0支持了Lead和Lag分析函数，虽然可以大幅提高同、环比计算的效率，但仍然需要编写SQL语句处理。

2.1计算同比：

selectt2.年份,t2.月份,concat(round((t2.数量-t1.数量)/t1.数量,2)*100,'%') as同比 from(SELECTyear(销售时间) as年份,month(销售时间) as月份,sum(销售数量) as数量 fromsales groupbyyear(销售时间),month(销售时间) orderbyyear(销售时间) desc, month(销售时间) desc) t1 ,(SELECTyear(销售时间) as年份,month(销售时间) as月份,sum(销售数量) as数量 fromsales groupbyyear(销售时间),month(销售时间) orderbyyear(销售时间) desc, month(销售时间) desc) t2 wheret1.年份=t2.年份-1andt1.月份=t2.月份

年份	月份	同比
2020	6	-53.00%
2020	5	53.00%

2.2计算环比：

SELECTmm, CONCAT(ROUND(IFNULL((xl - first_xl) / first_xl * 100, 2), 0), '%') AS环比 FROM(SELECTmm, xl, lead(xl, 1) over(ORDERBYmm DESC) ASfirst_xl FROM(SELECTDATE_FORMAT(销售时间, '%Y-%m') ASmm, sum(销售数量) ASxl FROMsales GROUPBYDATE_FORMAT(销售时间, '%Y-%m')) t) a

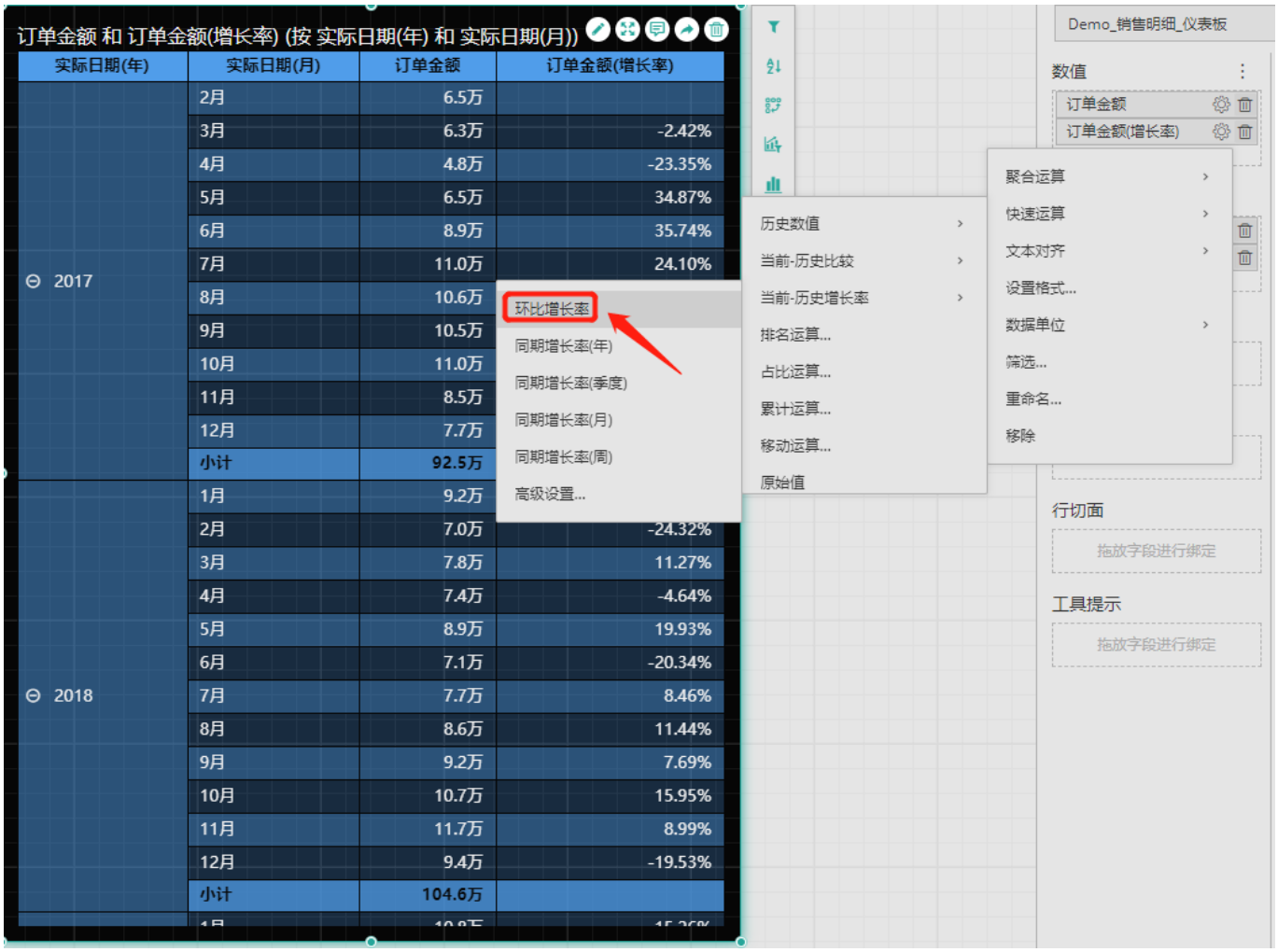
在SqlServer2008R2和Oracle10g之后，都提供了Lag和Lead分析函数。具体的计算逻辑和用法与上述MySQL8.0类似。

3、使用BI工具的计算引擎

针对此类复杂的计算场景，商业智能BI数据分析工具提供了更加高效的解决方案。以Wyn Enterprise嵌入式商业智能软件为例，其内置的wax分析表达式和快速计算引擎，提供直接实现同比、环比等复杂计算的能力，而不再需要写复杂冗长的SQL。

3.1 使用内置的同比、环比快速计算功能

同比、环比等计算一般是BI工具的标准功能，我们可以直接通过设置实现。



3.2 使用数据分析表达式

如果内置快速计算无法满足要求，还可以通过分析表达式实现更复杂的计算。分析表达式是一种更加灵活、强大的数据计算方式，通过丰富的函数，用户可以像Excel公式一样自由组合，实现更加强大的分析能力。分析表达式基于数据模型进行业务计算，以一些定义好的函数运用正确的语法来完成某个复杂的业务逻辑计算。这样可以使用户更灵活地使用数据，最大限度的利用数据。

通过对比SQL和BI数据分析工具在处理同比、环比等复杂计算中的差异，我们可以发现，还是专业的工具在数据计算和处理能力上要更加便捷。以后在工作中，如果有类似的分析计算需求，选择BI分析工具来处理是再合适不过的了。