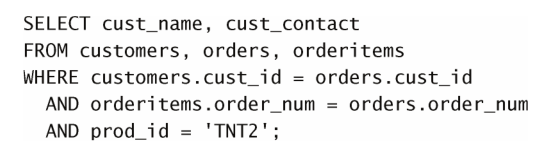
第22章：使用视图

1. **为什么要使用视图**

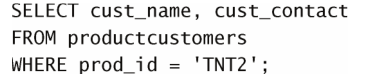
**\*视图是虚拟的表，与包含数据的表不同，它只包含使用时动态检索数据的查询。**

**第15章中用下面的SELECT语句从3个表中检索数据：**



**此查询用来检索订购了某个特定产品(prod\_id=’TNT2’)的客户姓名与联系方式。**

**现在，假如可以把整个查询包装成一个名为 productcustomers 的虚拟表，则可以如下轻松地检索出相同的数据：**



**其中productcustomers 是一个视图，作为视图，它不包含表中应该有的任何列或数据，它包含的是一个SQL查询（与上面用以正确联结表的相同的查询）。**

1. **重用SQL语句。**
2. **简化复杂的SQL操作。在编写查询后，可以方便地重用它而不必知道它的基本查询细节。**
3. **使用表的组成部分而不是整个表。**
4. **保护数据。可以给用户授予表的特定部分的访问权限而不是整个表的访问权限。**
5. **更改数据格式和表示。视图可返回与底层表的表示和格式不同的数据。**

**注意：**

**性能问题：因为视图不包含数据，所以每次使用视图时，都必须处理查询执行时所需的任一个检索。如果你用多个联结和过滤创建了复杂的视图或者嵌套了视图，可能会发现性能下降得很厉害。因此，在部署使用了大量视图的应用前，应该进行测试。**

1. **MySQL中视图和表的区别与联系**

**1）区别：**

1. **视图是已经编译好的sql语句，而表不是。**
2. **视图没有实际的物理记录，而表有。**
3. **表是内容，视图是窗口。**
4. **表只用物理空间而视图不占用物理空间，视图只是逻辑概念的存在，表可以及时地对它进行修改，但视图只能由创建的语句来修改。**
5. **表是内模式，视图是外模式。**
6. **视图是查看数据表的一种方法，可以查询数据表中某些字段构成的数据，只是一些SQL语句的集合。从安全的角度说，视图可以不给用户接触数据表，从而不知道表结构。**
7. **表属于全局模式中的表，是实表；视图属于局部模式的表，是虚表。**
8. **视图的建立和删除只影响视图本身，不影响对应的基本表。**
9. **定义视图并执行视图与基于派生表的查询是有区别的。视图一旦定义，其将永远保存在数据字典中，之后的所有查询都可以直接使用它，而派生表只是在语句执行时临时定义，语句执行后该定义即被删除。**
10. **由于视图是不实际存储数据的虚表，因此对它的更新最终要转化为对基本表的更新。（也是通过视图消解）**
11. **联系：**
12. **视图是在基本表之上建立的表，它的结构（即所定义的列）和内容（即所有记录）都来自基本表，它依据基本表存在而存在。**
13. **一个视图可以对应一个基本表，也可以对应多个基本表。**
14. **视图是基本表的抽象和在逻辑意义上建立的新关系。**
15. **视图的规则与限制**

** 1）与表一样，视图必须唯一命名（不能给视图取与别的视图或表相同的名字）。**

** 2）对于可以创建的视图数目没有限制。**

** 3）为了创建视图，必须具有足够的访问权限。这些限制通常由数据库管理人员授予。**

** 4）视图可以嵌套，即可以利用从其他视图中检索数据的查询来构造一个视图。**

** 5）ORDER BY 可以用在视图中，但如果从该视图检索数据 SELECT 中也含有 ORDER BY ，那么该视图中的 ORDER BY 将被覆盖。**

** 6）视图不能索引，也不能有关联的触发器或默认值。**

** 7）视图可以和表一起使用。例如，编写一条联结表和视图的 SELECT**

**语句。**

1. **可以在一个视图上再定义新的视图，但对视图的更新（增、删、改）操作则有一定的限制。**
2. **视图定义中的子查询可以是任意的SELECT语句，是否可以含有ORDER BY子句和DISTINCT 短语，则取决于具体系统的实现。**
3. **WITH CHECK OPTION表示对视图进行增、删、改操作时要保证更新、插入或删除的行满足视图定义中的谓词条件，即子查询中的条件表达式。**
4. **组成视图的属性列名或者全部省略或者全部指定，没有第三种选择。如果省略，则隐含该视图由子查询中SELECT子句目标列中的所有字段组成。但在下列三种情况下必须明确指定组成视图的所有列名：**
5. **某个目标列不是单纯的属性名，而是聚集函数或列表达式；**
6. **多表连接时选出了几个同名列作为视图的字段；**
7. **需要在视图中为某个列启用新的更合适的名字（别名）。**

**12）在关系数据库中，并不是所有的视图都是可以更新的，因为不能唯一地有意义地转换成对相应的基本表的更新。一般地，行列子集视图是可以更新的。（不同的系统有不同的规定）**

1. **视图的作用（P129，王珊《数据库系统概论》）**
2. **提高了重用性，就像一个函数。**
3. **对数据库重构，却不影响程序的运行。**
4. **提高了安全性能。可以对不同的用户，对机密数据设定不同的视图以安全保护。**
5. **让想要查询的数据更加清晰，想要什么样的数据，就创建什么样的视图。**
6. **由基本数据经过各种计算派生出的数据一般是不需要存储的，由于视图中的数据并不实际存储，所以定义视图时可以根据应用的需要设置一些派生属性列（虚拟列），带虚拟列的视图也称为带表达式的视图。**
7. **可以用带有聚集函数和GROUP BY子句的查询来定义视图（分组视图）。**
8. **最好在修改基本表之后删除由该基本表导出的视图，然后重建这个视图。**
9. **视图能简化用户的操作。**
10. **视图能使用户以多种角度看待同一数据。**
11. **视图对重构数据库提供了一定程度的逻辑独立性。**
12. **如何使用视图**

**CREATE [OR REPLACE] [ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}]**

**VIEW view\_name [(column\_list)]**

**AS select\_statement**

**[WITH [CASCADED | LOCAL] CHECK OPTION]**

**注：**

1. **OR REPLACE：表示替换已有视图**
2. **ALGORITHM：表示视图选择算法，默认算法是UNDEFINED(未定义的)：MySQL自动选择要使用的算法 ；merge合并；temptable临时表**
3. **select\_statement：表示select语句**
4. **[WITH [CASCADED | LOCAL] CHECK OPTION]：表示视图在更新时保证在视图的权限范围之内**

**cascade是默认值，表示更新视图的时候，要满足视图和表的相关条件**

**local表示更新视图的时候，要满足该视图定义的一个条件即可**

**TIPS：**

**推荐使用WHIT [CASCADED|LOCAL] CHECK OPTION选项，可以保证数据的安全性**

**基本格式：**

**create view <视图名称>[(column\_list)]**

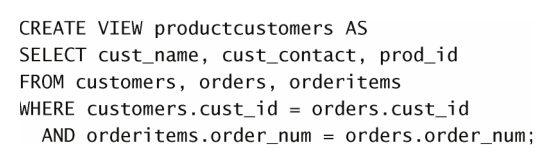
**as select语句**

**with check option;**

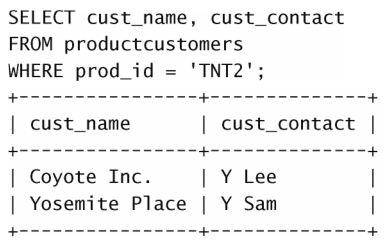
1. **视图用 CREATE VIEW 语句来创建。**
2. **使用 SHOW CREATE VIEW viewname；来查看创建视图的语句。**
3. **用 DROP 删除视图，其语法为 DROP VIEW viewname。**
4. **更新视图时，可以先用DROP再用CREATE，也可以直接用CREATE OR REPLACE VIEW。 如果要更新的视图不存在，则第2条更新语句会创建一个视图；如果要更新的视图存在，则第2条更新语句会替换原有视图。**
5. **Create view语句的结果只是把视图的定义存入数据字典，并不执行其中的SELECT语句。只是在对视图查询时，才按视图的定义从基本表中将数据查出。**
6. **执行对视图的查询时，首先进行有效性检查，检查查询中涉及的表、视图等是否存在。如若存在，则从数据字典中取出视图的定义，把定义中的子查询与用户的查询结合起来，转换成等价的对基本表的查询，然后再执行修正了的查询。这一转换过程称为视图消解。**

**具体用法：**

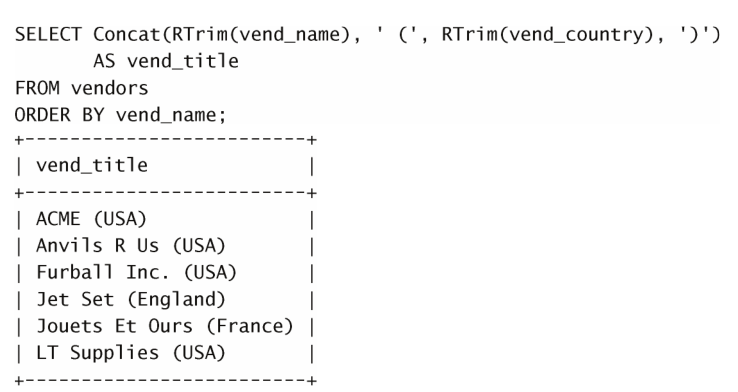
1. **利用视图简化复杂的查询（联结）**

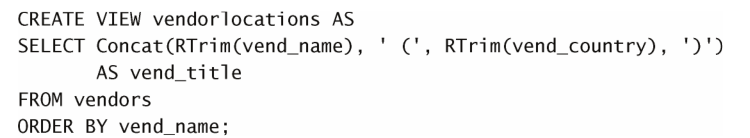


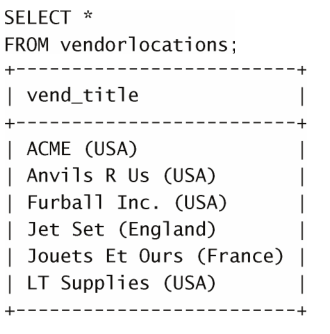
**如果执行SELECT \* FROM productcustomers ，将列出订购了任意产品的客户。**



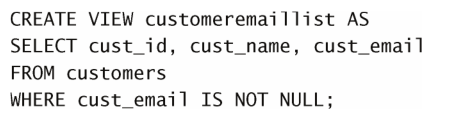
1. **用视图可以重新格式化检索出的数据**

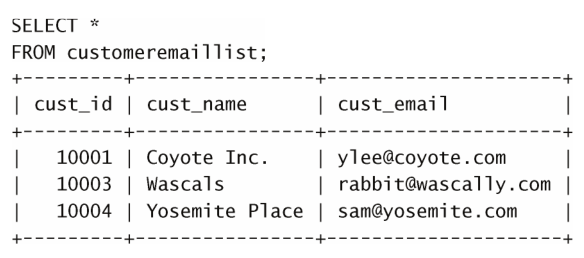






1. **用视图过滤不必要的数据**

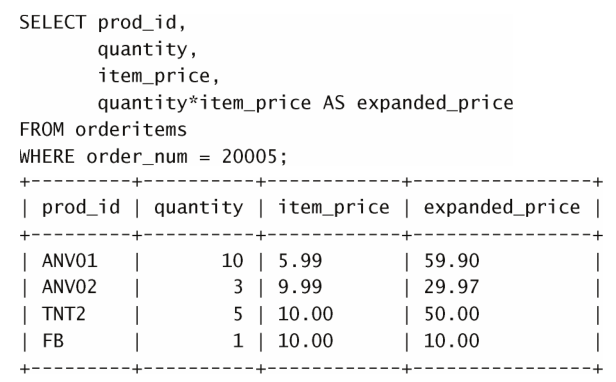


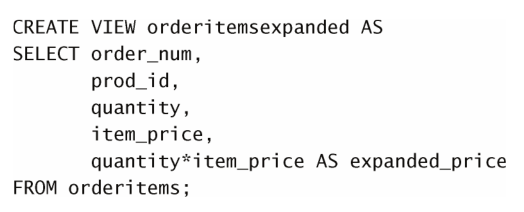


**WHERE 子句与 WHERE 子句：**

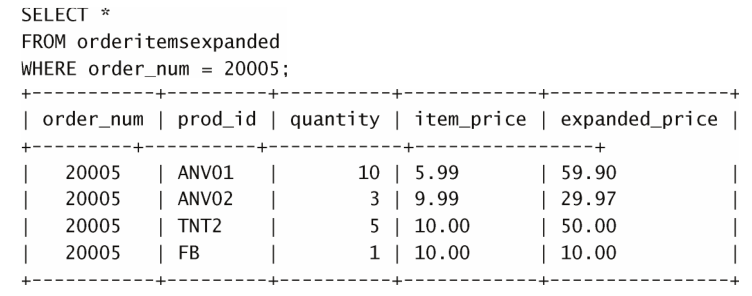
**如果从视图检索数据时使用了一条WHERE 子句，则两组子句（一组在视图中，另一组是传递给视图的）将自动组合。**

1. **使用视图结合计算字段进行统计分析**





**为检索订单 20005 的详细内容（上面的输出），如下进行：**



1. **更新视图**

**通常，视图是可更新的（即，可以对它们使用 INSERT 、 UPDATE 和DELETE ）。更新一个视图将更新其基表（视图本身没有数据）。如果你对视图增加或删除行，实际上是对其基表增加或删除行。**

**但是，并非所有视图都是可更新的。基本上可以说，如果MySQL不能正确地确定被更新的基数据，则不允许更新（包括插入和删除）。**

**这实际上意味着，如果视图定义中有以下操作，则不能进行视图的更新：**

** 1）分组（使用 GROUP BY 和 HAVING ）；**

**2）联结；**

**3）子查询；**

** 4）并；**

** 5）聚集函数（ Min() 、 Count() 、 Sum() 等）；**

** 6）DISTINCT；**

** 7）导出（计算）列；**

1. **引用FROM子句中的不可更新视图；**

**9）仅引用文字值；**

**10）对基表的任何列的多次引用。**

**将视图用于检索：**

**一般，应该将视图用于检索（ SELECT 语句）而不用于更新（ INSERT 、 UPDATE 和 DELETE ）。**

附关系图：

