项目5 MySQL 8视图、索引及其应用

5.1 项目描述

在图书馆查找需要的书，在《新华字典》里查找某个汉字，在十四亿多的中国人口数据中查找某个人等问题中，若没有索引目录，查找势必难如登天。而数据库存储数据的主要目的是方便用户查询。在 MySQL 8 中，为提高查询速度，需对数据表建立相应的索引，此外，为了能够根据用户的不同需求简化数据的查询操作，需要定义和使用视图。

本项目旨在通过实际操作，让读者学会 MySQL 8 的视图以及索引的使用，具体任务包括如下三方面。

（1）MySQL 8 中视图的创建、查询、更新、修改、删除。

（2）MySQL 8 中索引的创建、删除。

（3）MySQL 8 中数据完整性的实现方法。

5.2 任务解析

在实际应用中，根据用户的不同需求，在物理数据库上定义用户对数据库所要求的数据结构，这种根据用户观点所定义的数据结构就是视图。在 MySQL 8 中，视图一经定义，就可以像表一样进行查询、修改、删除和更新等操作。也就是说，使用视图可有效地简化数据查询操作。本项目将详细介绍视图的概念、功能、创建方法，及查询、更新、修改等操作。

索引是对数据表中的一列或多列的值进行排序的一种结构，使用索引可提高数据库中特定数据的查询速度。不合理的索引或者缺少索引都会对数据库和应用程序的性能造成影响。本项目将详细介绍索引的含义、作用与分类，索引的各种创建方法，索引与约束的关系，以及各种数据完整性的实现方法。

5.3 相关知识

5.3.1 MySQL 8 数据库视图

1. 视图概述

数据库中的视图是从一个或多个表中导出、供用户使用的虚拟表。视图是数据库用户使用数据库时的虚拟表。视图与表有很多相似的地方，视图也是由若干字段以及若干记录

构成的，可以作为 SELECT 语句的数据源，甚至在某些特定条件下能够通过视图对表进行更新操作。然而，视图中的数据并不像表、索引那样需要占用存储空间，视图中保存的仅仅是一条 SELECT 语句，其源数据都来自数据库表，数据库表称为基本表或基表，视图称为虚表。基表的数据发生变化时，虚表的数据也会随之变化。

MySQL 8 实现了视图功能（包括可更新视图），且 MySQL 5 及以上版本提供了二进制版的视图功能。MySQL 8[视图允许用户像单个表那样访问一组关系（表），也能限制对行的访问（特定表的子集）]对于列控制的访问，可使用 MySQL 服务器中的高级权限系统。

与直接从数据库表中提取数据相比，视图的特点如下。

- 使操作简单化：使用视图可以简化数据的查询操作，对于经常使用但结构复杂的 SELECT 语句，建议将其封装为一个视图。
- 避免数据冗余：由于视图保存的是 SELECT 语句，所有的数据保存在数据库表中，因此，这样就可以用一个或多个表派生出来多个视图，为相应的应用程序提供服务的同时，还避免了数据冗余。
- 增强数据安全性：同一个数据库表可以创建不同的视图，并为不同的用户分配不同的视图；可以实现不同的用户只查询或修改与之相对应的数据，继而增强了数据的安全性。
- 逻辑数据独立性：若没有视图，应用程序一定是建立在数据库表上的。有了视图后，应用程序就可以建立在视图之上，从而使应用程序和数据库表结构在一定程度上逻辑分离。

视图在以下两个方面使应用程序和数据逻辑独立。

- 使用视图能够向应用程序屏蔽表的结构，即使表结构发生变化（例如，表的字段名发生变化），此时只需重新定义视图或者修改视图的定义，无须修改应用程序就可以解决应用程序相应的问题。
- 使用视图可以向数据库中的表屏蔽应用程序，此时即使应用程序发生了变化，只需重新定义视图或者修改视图的定义，无须修改数据库表结构就可使应用程序正常运行。

2. 创建视图

创建视图需要有 CREATE VIEW 的权限，并且对于查询涉及的列有 SELECT 权限。

视图中包含了 SELECT 语句查询的结果，因此，视图的创建基于 SELECT 语句和已存在的数据表，视图可以建立在一张表上，也可以建立在多张表上。

创建视图的语法格式如下：

```
CREATE
[OR REPLACE]
[ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}]
VIEW view_name [(column_list)]
AS select_statement
[WITH [CASCADED | LOCAL] CHECK OPTION]
```

参数说明如下。

view_name：视图名。

column_list：为视图的列定义明确的名称，可使用可选的 column_list 子句，列出由逗号隔开的列名。column_list 中的名称数目必须等于 SELECT 语句检索出的列数，但是在使用与源表或视图中相同的列名时可以省略 column_list。

OR REPLACE：给定了 OR REPLACE 子句，语句能够替换已有的同名视图。

ALGORITHM 子句：可选的 ALGORITHM 子句是对标准 SQL 语句的 MySQL 扩展，规定了 MySQL 的算法，算法会影响 MySQL 处理视图的方式。ALGORITHM 可取 3 个值：UNDEFINED、MERGE 或 TEMPTABLE。如果没有 ALGORITHM 子句，默认算法是 UNDEFINED(未定义的)。如果指定了 MERGE 选项，会将引用视图的语句的文本与视图定义结合起来，使得视图定义的某一部分取代语句中的对应部分。MERGE 算法要求视图中的行和基表中的行一一对应，如果不具有该关系，必须使用临时表取而代之。如果指定了 TEMPTABLE 选项，视图的结果将被放置在临时表中，然后使用它执行语句。

select_statement：用来创建视图的 SELECT 语句，可在 SELECT 语句中查询多个表或视图。但对 SELECT 语句有以下限制。

- 定义视图的用户必须对所参照的表或视图有查询(即可执行 SELECT 语句)权限。
- 不能包含 FROM 子句中的子查询。
- 不能引用系统或用户变量。
- 不能引用预处理语句参数。
- 在定义中引用的表或视图必须存在。
- 若引用的不是当前数据库的表或视图时，要在表或视图前加上数据库的名称。
- 在视图定义中允许使用 ORDER BY 语句，如果从特定视图进行了选择，而该视图使用了自己的 ORDER BY 语句，则视图定义中的 ORDER BY 语句将被忽略。
- 对于 SELECT 语句中的其他选项或子句，若视图中也包含了这些选项，则效果未定义。例如，如果在视图定义中包含 LIMIT 子句，而 SELECT 语句使用了自己的 LIMIT 子句，则 MySQL 对使用的 LIMIT 语句未做定义。

WITH CHECK OPTION：指出在可更新视图上所进行的修改都要符合 select_statement 所规定的限制条件，这样可以确保数据修改后，可以通过视图看到修改的数据。当视图是根据其他视图定义时，WITH CHECK OPTION 给出两个参数：LOCAL 和 CASCADED，它们决定了检查测试的范围。LOCAL 关键字使 CHECK OPTION 只对定义的视图进行检查，CASCADED 则会对所有视图进行检查。如果未给定任一关键字，则默认值为 CASCADED。

创建视图的指导原则如下。

- 只能在当前数据库中创建视图。
- 视图名称应遵循标识符的命名规则。
- 可以基于其他视图创建视图。
- 不能将默认值、规则和触发器与视图相关联。
- 不能为视图建立索引。
- 创建视图时不能使用临时表。
- 即使表被删除，视图定义仍将保留。
- 定义视图的查询不能包含以下语句：COMPUTE 子句、COMPUTE BY 子句、INTO 关键字。

3. 查看视图

查看视图是查看数据库中已创建的视图。用 MySQL Workbench 查看数据库视图的办法是在 SCHEMAS 中打开数据库，选择 Views 即可。

另外，在 MySQL 8 控制台下，也可通过 DESCRIBE（简写为 DESC）命令查看指定的视图。

4. 更新视图

更新视图是指通过视图来插入、更新或删除表中的数据，因为视图就是一个虚拟表，表中没有数据。视图更新时都是转到基本表上进行更新的，如果对视图增加或删除记录，实际上就是对其基本表增加或删除记录。

要通过视图更新基本表的数据，必须保证视图是可更新视图，即可以在 INSET、UPDATE 或 DELETE 等语句中使用它们。对于可更新的视图，在视图中的行和基表中的行之间必须具有一一对应的关系。

还有一些特定的其他结构，若使用这类结构，视图将变得不可更新。如果视图包含聚合函数、DISTINCT 关键字、GROUP BY 子句、ORDER BY 子句、HAVING 子句、UNION 运算符、位于选择列表中的子查询、FROM 子句中包含多个表、SELECT 语句中引用了不可更新视图、WHERE 子句中的子查询、引用 FROM 子句中的表或 ALGORITHM 选项指定为 TEMPTABLE（使用临时表总会使视图成为不可更新的）中的任何一种，那么它就是不可更新的。

更新视图的三个命令：INSERT、UPDATE 和 DELETE。

5. 修改视图

修改视图是指修改 MySQL 8 数据库中存在的视图，当基本表的某些字段发生变化时，可以通过修改视图来保持与基本表的一致性。MySQL 8 中通过 CREATE OR REPLACE VIEW 语句、ALTER 语句实现对视图的修改。

可以看到，修改视图的 CREATE OR REPLACE VIEW 语句和创建视图的语句是完全一样的。需要注意的是，在执行 CREATE OR REPLACE VIEW 语句时，若指定的视图名称已存在时，则该语句对视图进行修改；当视图不存在时创建视图。

在 MySQL 8 中，用 ALTER 语句修改视图的方法如下：

```
ALTER [ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}]
   VIEW view_name [(column_list)]
   AS select_statement
   [WITH [CASCADED | LOCAL] CHECK OPTION]
```

6. 删除视图

当不再需要视图时，可以将其删除，删除一个或多个视图可以使用 DROP VIEW 语句。语法如下：

```
DROP VIEW [IF EXISTS]
view_name [, view_name] …
[RESTRICT | CASCADE]
```

其中，view_name 是视图名，声明了 IF EXISTS，若视图不存在的话，就不会出现错误提示信息。也可以声明 RESTRICT 和 CASCADE，但它们没什么影响。

5.3.2 MySQL 8 索引

创建数据库表时，初学者通常仅仅关注数据表有哪些字段、字段的数据类型及约束条件这些信息，很容易忽视数据库表中的另一个重要概念索引。

1. 索引简介

想象一下《现代汉语词典》的使用方法，就可以理解索引的重要性。《现代汉语词典》全书将近 1800 页，收录汉字超过 1.3 万个，如何在众多汉字中找到某个字（如翔）？从《现代汉语词典》的第一页开始逐页、逐字查找，直到查找到含有“翔”字的那一页，相信读者不会这样做。词典提供了音节表，音节表将汉字拼音 xiang 编入其中，并且音节表按 a 到 z 的顺序排列，故而读者可以轻松地在音节表中找到 xiang 1488，然后再从 1488 页开始逐字查找，这样可以快速地检索到“翔”字。音节表就是《现代汉语词典》的一个索引，其中，音节表中的 xiang 是索引的关键字，该关键字的值必须来自词典正文中的 xiang（或者说词典正文中 xiang 的复制），索引中的 1488 是“数据”所在的起始页。数据库表中存储的数据通常比《现代汉语词典》收录的汉字多得多，在没有索引的词典中查找某个字对读者而言变得举步维艰，同样，在没有索引的数据库表中寻找需要的数据对于数据库用户而言更是如同大海捞针。

- 索引的本质是什么？本质上，索引是数据库表中某字段值的复制，该字段称为索引的关键字，索引的目的是提高查询效率。
- MySQL 数据库中，数据是如何检索的？答案就是 MySQL 在检索表中的数据时，先按照索引关键字的值在数据库中进行查找，若能够查到，则可以直接定位到数据所在的起始页；如果没有查到，就只能全表扫描查找想要的数据了。
- 一个数据库表只能创建一个索引吗？当然不是。想象一下《现代汉语词典》，除了将汉语拼音编入音节表实现汉字的检索功能外，还将所有汉字的偏旁部首编入部首检字表实现汉字的检索功能，部首检字表是《现代汉语词典》的另一个索引。同样对于 MySQL 数据库表而言，一个数据库表也可以创建多个索引。

在 MySQL 8 中，建立索引的主要作用如下。

- 通过创建唯一索引，可以保证数据库表中每行数据的唯一性。
- 可以大大提高数据的查询速度，这也是创建索引的主要原因。
- 在实现数据的参照完整性方面，可以加速表与表之间的连接。
- 在使用分组和排序子句进行数据查询时，也可以显著地减少查询中分组和排序的时间。

在 MySQL 8 数据库中，索引的设计不合理或者缺少索引都会对数据库和应用程序的性能造成影响。高效的索引是获得良好性能非常重要的一环。设计索引时，应该考虑以下原则。

- 索引并非越多越好。一个表中如果建立了大量的索引，不仅占用磁盘空间，还会影响 INSERT、DELETE、UPDATE 等语句的性能，因为当表中数据更改的同时，索引也会进行调整和更新。
- 数据量小的数据表最好不要使用索引。由于数据量小，查询花费的时间可能比遍历索引的时间还要多，索引可能不会产生优化效果。
- 避免对经常更新的数据表建立过多的索引，并且索引中的列尽可能少。但对经常用于查询的字段应该建立索引，且要避免添加不必要的字段。

- 在条件表达式中经常用到在不同值较多的列上建立索引，在不同值少的列上不要建立索引。
- 当唯一性是某种数据本身的特性时，指定唯一索引，提高查询速度。
- 在频繁进行排序或分组的列上建立索引。

2. 索引的分类

MySQL 8 的索引可以分成以下几类。

- 普通索引和唯一索引：普通索引是 MySQL 8 的基本索引类型，允许在定义索引的列中插入重复值和空值；唯一索引，索引列的值必须唯一，但允许值为空值。
- 单列索引和组合索引：单列索引即一个索引只包含单个列，一个表可以有多个单列索引；组合索引指在表的多个字段的组合上创建的索引，只有在查询条件中使用了这些字段的左边字段时，索引才会被使用。使用组合索引时遵循最左前缀组合。
- 全文索引：全文索引类型为 FULLTEXT，在定义索引的列上支持组的全文查找，允许在这些索引列中插入重复值和空值。全文索引可以在 CHAR、VARCHAR 或者 TEXT 类型的列上创建。MySQL 8 中只有 MySQL 存储引擎支持全文索引。
- 空间索引：空间索引是对空间数据类型的字段建立的索引。MySQL 8 中的空间数据类型有四种，分别是：GEOMETRY、POINT、LINESTRING 和 POLYGON。

3. 创建索引

MySQL 8 支持多种方法在单个或多个数据表的列上创建索引：在创建表的定义语句 CREATE TABLE 中指定索引列；使用 ALTER TABLE 语句在存在的表上创建索引；使用 CREATE INDEX 语句在已存在的表上添加索引。

在创建表时创建索引：使用 CREATE TABLE 创建表时，既可以定义列的数据类型，也可以定义主键约束、外键约束或者唯一性约束，而不论创建哪种约束，在定义约束的同时相当于在指定的列上创建了一个索引。创建表时创建索引的基本语法格式如下：

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    [ ( [column_definition] , … | [index_definition] ) ]
    [table_option] [select_statement];
```

其中，index_definition 为索引项：

```
  [CONSTRAINT [symbol]] PRIMARY KEY [index_type] (index_col_name, … )      /* 主键 */
| {INDEX | KEY} [index_name] [index_type] (index_col_name, … )             /* 索引 */
| [CONSTRAINT [symbol]] UNIQUE [INDEX] [index_name] [index_type] (index_col_name, … )
/* 唯一性索引 */
| [FULLTEXT | SPATIAL] [INDEX] [index_name] (index_col_name, … )           /* 全文索引 */
| [CONSTRAINT [symbol]] FOREIGN KEY [index_name] (index_col_name, … ) [reference_definition]
/* 外键 */
```

说明：KEY 通常是 INDEX 的同义词。在定义数据表列选项时，也可以将某列定义为 PRIMARY KEY，但是当主键是由多个列组成的多列索引时，定义列时无法定义此主键，必须在语句最后加上一个 PRIMARY KEY(col_name,…)子句。

在 MySQL 8 中，用 CREATE INDEX 语句创建索引的语法格式如下：

```
CREATE [UNIQUE] [CLUSTER] INDEX <索引名>
ON <表名>(<列名>[<次序>][,<列名>[<次序>]]…);
```

其中：

- <表名>：要创建索引的基本表的名字。
- <列名>：可以建立在该表的一列或多列上，各列名之间用逗号分隔。
- <次序>：指定索引值的排列次序。升序(ASC)、降序(DESC)、默认值为 ASC。
- UNIQUE：此索引的每个索引值只对应唯一的数据记录。
- CLUSTER：表示要建立的索引是聚簇索引。

4. 删除索引

如果某些索引降低了数据库的性能，或者根本没有必要使用该索引，可以考虑删除该索引。MySQL 8 中使用 ALTER TABLE 或者 DROP INDEX 语句删除索引。DROP INDEX 语句在内部被映射到一个 ALTER TABLE 语句中。

使用 DROP INDEX 语句删除索引的语法格式如下：

```
DROP INDEX index_name ON tbl_name;
```

DROP INDEX 语句的语法相对简单，index_name 为要删除的索引名，tbl_name 为索引所在的数据表。

需要注意的是：DROP INDEX 语句不适用于用主键约束或唯一性约束创建的索引。DROP INDEX 也不能用于删除系统表的索引。

5. 索引与约束

MySQL 8 中表的索引与约束之间存在怎样的关系？约束分为主键约束、唯一性约束、默认值约束、检查约束、非空约束和外键约束。其中主键约束、唯一性约束以及外键约束与索引的联系较为紧密。

约束主要用于保证业务逻辑操作数据库时数据的完整性，而索引则是将关键字数据以某种数据结构的方式存储到外存，用于提升数据的检索性能。约束是逻辑层面的概念，而索引既有逻辑上的概念，更是一种物理存储方式，且事实存在，需要占用一定的存储空间。

对于一个 MySQL 8 数据库表而言，主键约束、唯一性约束以及外键约束是基于索引实现的。因此，对于主键约束、唯一性约束以及外键约束，创建约束的同时，会自动创建一个同名索引。

MySQL 8 数据库中删除了唯一性索引，其对应的唯一性约束也将自动删除。

5.4 任务实施

5.4.1 MySQL 8 数据库视图操作

【例 5.1】 假设 MySQL 8 中的当前数据库是 jwgl，创建该数据库上的视图 jwgl_view1，要求该视图中包括计算机科学与技术专业学生 2021—2022 学年第 2 学期的选课信息，要求呈现学生的学号、姓名、专业、选课名称信息。

MySQL Workbench 中的操作命令如下：

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `jwgl_view1` AS
    select
        `student`.`Sno` AS `Sno`,
        `student`.`Sname` AS `Sname`,
        `student`.`Smajor` AS `Smajor`,
        `select_course`.`course_number_select` AS `course_number_select`
    from
        (`student`
        join `select_course`)
    where
        ((`student`.`Sno` = `select_course`.`Sno_select`)
            and (`student`.`Smajor` = '计算机科学与技术')) WITH CASCADED CHECK OPTION
```

在 MySQL Workbench 中的操作结果如图 5.1 所示。

图 5.1　视图 jwgl_view1 的查看结果

【例 5.2】 用 MySQL 8 在例 5.1 基础上创建新的视图 jwgl_view2，要求该视图以中文形式显示学号、姓名、课程号。

MySQL Workbench 中的操作命令如下：

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `jwgl_view2` AS
    SELECT
        `jwgl_view1`.`Sno` AS `学号`,
```

```
        `jwgl_view1`.`Sname` AS `姓名`,
        `jwgl_view1`.`course_number_select` AS `所选课程的课程号`
    FROM
        `jwgl_view1`
```

从例 5.2 可以看出，视图定义后，就可以如同查询基本表那样对该视图进行查询。

需要说明的是：在使用视图查询时，若其关联的基本表中添加了新的字段，则该视图将不包含该新字段；同时，如果与视图相关联的表或视图被删除，则该视图将不能再继续使用。

【例 5.3】 查看 MySQL 8 的视图信息。查看例 5.1 和例 5.2 创建的视图。

操作方法：用 MySQL Workbench 连接 MySQL 8 本地服务器的 jwgl 数据库，在 SCHEMAS 中选择数据库 jwgl，再选择 Views 即可，如图 5.2 所示。

图 5.2 用 MySQL Workbench 查看数据库视图

【例 5.4】 在 MySQL 8 的控制台下，用 DESCRIBE 命令查看例 5.1 和例 5.2 创建的视图 jwgl_view1、jwgl_view2 分别如图 5.3 和图 5.4 所示。

【例 5.5】 在 MySQL 8 的控制台下用 SHOW CREATE VIEW 语句可查看视图的详细信息。基于该语句查看 jwgl_view2 的视图信息如图 5.5 所示。

【例 5.6】 用 INSERT 语句通过视图向基本表插入数据。在 MySQL 8 的数据库 jwgl 中创建视图 jwgl_view3，要求该视图中包含物联网应用技术专业学生的学号、姓名信息；显示视图 jwgl_view3 内容；向 jwgl_view3 视图中插入一条记录(20171606050107，杨超)并验证。

操作步骤 1：在 MySQL Workbench 中创建视图 jwgl_view3，实现代码如图 5.6 所示。

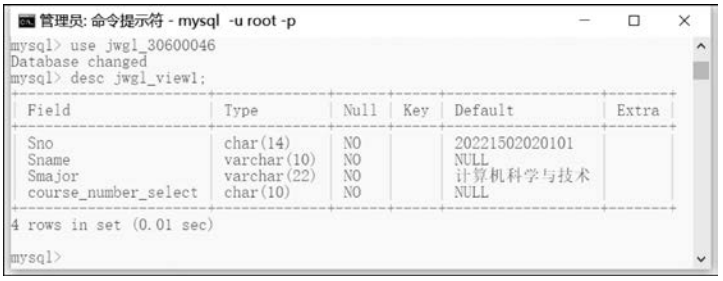

图 5.3　用 DESCRIBE 命令在控制台下查看 jwgl_view1 视图信息

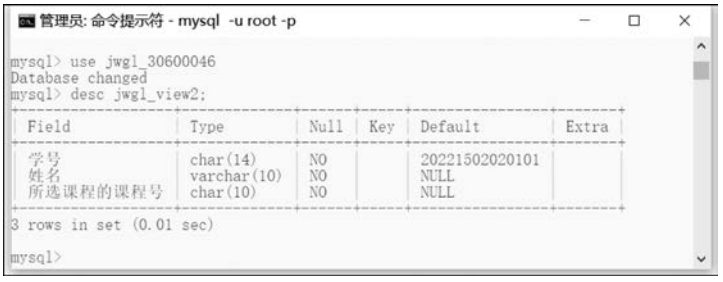

图 5.4　用 DESCRIBE 命令在控制台下查看 jwgl_view2 视图信息

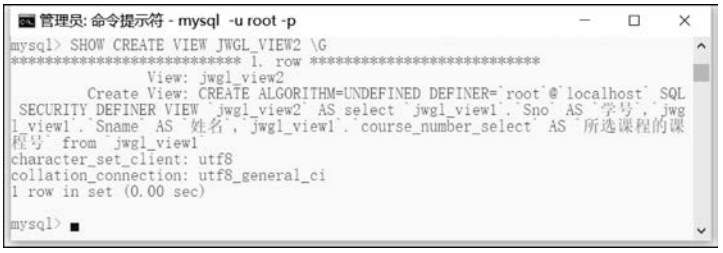

图 5.5　用 SHOW CREATE VIEW 语句在控制台下查看 jwgl_view2 视图信息

图 5.6　在 MySQL Workbench 中创建视图 jwgl_view3 的代码

操作步骤 2：在 MySQL Workbench 中显示视图 jwgl_view3 内容的代码如图 5.7 所示。

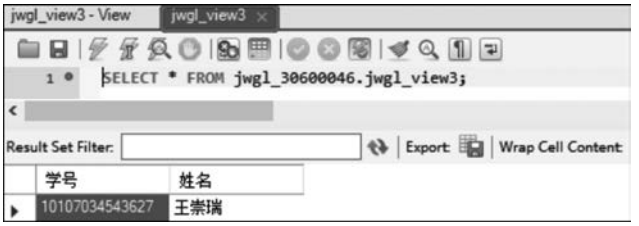

图 5.7　在 MySQL Workbench 中显示视图 jwgl_view3 内容的代码

操作步骤 3：基于 MySQL Workbench 在视图 jwgl_view3 中插入一条记录(20171606050107,杨超),实现代码如图 5.8 所示。

```
use jwgl_30600046;
INSERT INTO jwgl_view3 VALUES('20171606050107','杨超');
```

图 5.8　在 MySQL Workbench 中向视图 jwgl_view3 添加记录的代码

操作步骤 4：基于 MySQL Workbench 验证学生信息表中是否添加了学号、姓名分别是 20171606050107、“杨超”的新记录,实现代码及验证结果如图 5.9 所示。

图 5.9　用 MySQL Workbench 验证基础数据表记录变化的实现代码及验证结果

需要说明的是：这里通过 INSERT 语句通过视图成功地向基本表插入数据时,记录中性别 Ssex、出生日期 Sbirth 等字段的值用默认值填充。

除了 INSERT 外,MySQL 8 还允许使用 UPDATE、DELETE 命令对视图的内容进行更新操作,用法同 MySQL 8 用 UPDATE、DELETE 命令操作一般的数据表,此处不再赘述。

【例 5.7】 基于数据库 jwgl 创建临时用的视图 jwgl_view,显示该视图全部内容后用 DROP 命令删除该视图。

操作步骤 1：创建视图 jwgl_view,实现代码如图 5.10 所示。

操作步骤 2：显示视图 jwgl_view 的全部内容,实现代码如图 5.11 所示。

操作步骤 3：用 DROP 命令删除视图 jwgl_view,实现代码如下：

```
DROP VIEW jwgl.jwgl_view;
```

```
USE `jwgl_30600046`;
CREATE
    OR REPLACE ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `jwgl_view` AS
    select
        `class`.`Cno` AS `班级编号`,
        `class`.`Cname` AS `班级名称`,
        `class`.`Csunmebr` AS `班级人数`,
        `class`.`Sno` AS `班长学号`
    from
        `class`;
```

图 5.10　创建数据库 jwgl 的视图 jwgl_view 的代码

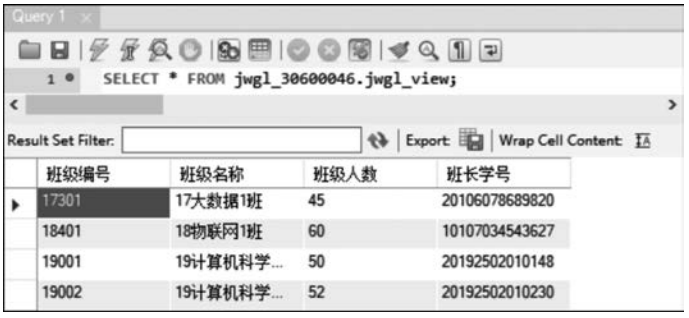

图 5.11　显示视图 jwgl_view 全部内容

5.4.2　MySQL 8 数据库索引操作

【例 5.8】　针对本项目前述的 MySQL 8 数据库 jwgl 的学生信息表创建唯一索引(索引名：Stusno_index)，并要求按照“学号”字段升序创建索引。

基于 MySQL Workbench 的操作命令如下：

```
CREATE UNIQUE INDEX Stusno_index ON jwgl.student(Sno);
```

【例 5.9】　将例 5.8 的索引名称修改为 Stusno_indexnew。

MySQL 8 的操作命令如下：

```
ALTER TABLE jwgl.student RENAME INDEX Stusno_index RENAME TO Stusno_indexnew;
```

补充说明：MySQL 5.7 及以下版本中的操作命令如下。

```
ALTER TABLE jwgl.student DROP INDEX Stusno_index;
ALTER TABLE jwgl.student ADD INDEX Stusno_indexnew(Sno);
```

5.5　任务小结

通过对“MySQL 8 视图、索引及其应用”项目的学习和训练，大家应该理解在 MySQL 8 中建立视图、索引的目的，操作方法及过程，同时掌握不再需要 MySQL 视图、索引时删除视图、索引的办法。

需要注意如下几点。

（1）视图是数据库中的虚拟表，它存储的是通过 SELECT 语句从其他表中整合而来的虚拟表。当其他表的内容改变时，视图的内容会随之改变，并且对视图的更新也会改变源表的内容。

（2）建立索引时，作为频繁查询条件的字段应该创建索引，唯一性太差的字段不适合建立索引，登录状态下更新非常频繁的字段不适合建立索引，不会出现在 WHERE 子句中的字段不该创建索引。

5.6 拓展提高

本项目重点讨论了 Windows 10 下 MySQL 8 视图、索引及其应用，通过实例着重介绍了 MySQL 8 视图、索引的常用操作方法，但 MySQL 8 如何让视图利用索引、索引的利弊、二级索引如何建立和使用等问题也是有必要了解的，请感兴趣的读者自行查阅相关资料。

自测与实验 5 MySQL 8 视图、索引及其应用

1. 实验目的

（1）验证 MySQL 8 的视图操作。

（2）验证 MySQL 8 的索引操作。

2. 实验环境

（1）PC 一台。

（2）MySQL 8 数据库管理系统。

3. 实验内容

（1）MySQL 8 视图的创建、查询、更新、修改、删除操作。

（2）MySQL 8 中索引的创建、删除。

4. 实验步骤

参照本项目的任务实施。