南京邮电大学

实 验 报 告

（ 2024 / 2025 学年 第 二 学期）

课程名称	数据库系统基础
实验名称	实验二：MySQL 进阶开发
实验时间	2025 年 5 月 7 日
指导单位	计算机学院、软件学院、网络空间安全学院
指导教师	黄楠

学生姓名	于明宏	班级学号	B23041011
学院(系)	计软网安院	专 业	信息安全

实 验 报 告

实验名称	实验二：MySQL 进阶开发		
实验类型	验证	实验学时	2

一、 实验目的和要求

（1）理解表记录的查询过程，并熟练掌握使用 Navicat for MySQL 进行表记录的查询操作，包括单表记录的查询、聚合函数的查询、多表连接查询以及子查询等；

（2）理解索引的概念及其作用，掌握索引的查看、创建、使用和删除方法；

（3）理解视图的概念及其作用，掌握视图的创建、查看、修改、删除及应用方法。

二、实验环境(实验设备)

硬件: 微型计算机

软件: Windows 操作系统、MySQL 5.6 或更高版本、Navicat for MySQL 15 或更高版本

三、实验原理及内容

1 表记录的查询

在数据库 studentinfo 中创建一个名为"Student1"的数据表，包括字段（id, name, age, grade），并插入一些示例数据，包括但不限于：(1, 'Alice', 18, 'A'), (2, 'Bob', 20, 'B'), (3, 'Charlie', 17, 'A'), (4, 'David', 19, 'B'), (5, 'Emily', 21, 'A')；

（1）请在 Student1 表中找出年级为"B"且出生于 2005 年（含）以后的学生记录，给出 SQL 语句并输出截图：

```
CREATE DATABASE IF NOT EXISTS studentinfo;
USE studentinfo;
CREATE TABLE IF NOT EXISTS Student1 (
    id INT PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    age INT,
    grade CHAR(1)
);
INSERT INTO Student1 (id, name, age, grade) VALUES
(1, 'Alice', 18, 'A'),
(2, 'Bob', 20, 'B'),
(3, 'Charlie', 17, 'A'),
(4, 'David', 19, 'B'),
```

```
(5, 'Emily', 21, 'A'),
(6, 'John', 16, 'A');
SELECT * FROM Student1 WHERE grade = 'B' AND age >=18;
```

（2）请使用聚合函数进行查询，计算每个年级学生的人数，给出 SQL 语句并输出截图：

```
SELECT grade, COUNT(name) AS number FROM Student1 GROUP BY grade;
```

（3）假设有另一个成绩表格"Grades"，包含字(student_id, math_grade, science_grade)，并插入一些示例数据，包括但不限于：(1, 90, 85), (2, 80, 92), (3, NULL, 88), (4, 95, NULL), (6, 85, 90)；请使用多表连接查询，找出每个学生及其对应的数学和科学成绩（包括没有成绩的学生），给出 SQL 语句并输出截图：

```
CREATE TABLE IF NOT EXISTS Grades (
    student_id INT PRIMARY KEY,
    math_grade INT,
    science_grade INT,
    FOREIGN KEY (student_id) REFERENCES Student1(id)
);
INSERT INTO Grades (student_id, math_grade, science_grade) VALUES
(1, 90, 85),
(2, 80, 92),
(3, NULL, 88),
(4, 95, NULL),
(6, 85, 90);
SELECT id, name, math_grade, Grades.science_grade FROM Student1 INNER JOIN Grades ON
Student1.id = Grades.student_id;
```

（4）请使用子查询，找出年龄大于所有年级平均年龄的学生记录，给出 SQL 语句并输出截图：

```
SELECT * FROM student1 WHERE age > (SELECT AVG(age) FROM student1);
```

2 索引的操作

在数据库 studentinfo 中创建一个名为"Student2"的表格，包括字段(id, name, age, department)，并插入一些示例数据，如下所示：(1, 'Alice', 18, 'Math'), (2, 'Bob', 20, 'Science'), (3, 'Charlie', 17, 'Math'), (4, 'David', 19, 'History'), (5, 'Emily', 21, 'Science');

（1）查看"Student2"表格的名为"idx_name"的唯一索引的详细信息，给出 SQL 语句并输出截图：

```
CREATE TABLE IF NOT EXISTS Student2 (
    id INT PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    age INT,
    department VARCHAR(50)
);
INSERT INTO Student2 (id, name, age, department) VALUES
(1, 'Alice', 18, 'Math'),
```

```
(2, 'Bob', 20, 'Science'),
(3, 'Charlie', 17, 'Math'),
(4, 'David', 19, 'History'),
(5, 'Emily', 21, 'Science');
SHOW INDEX FROM Student2 WHERE key_name = 'idx_name';
```

（2）创建组合唯一索引：请为"Student2"表格的"department"和"age"字段创建一个组合唯一索引，名称自拟，给出 SQL 语句并输出截图：

```
CREATE UNIQUE INDEX idx_department_age ON Student2 (department,age);
```

（3）使用组合唯一索引：使用适当的索引，查询科目为"Math"且年龄为 18 岁的学生记录，给出 SQL 语句并输出截图：

```
SELECT * FROM Student2 WHERE department = 'math' AND age = 18;
```

（4）删除索引：删除"Student2"表格的组合唯一索引，给出 SQL 语句并输出截图：

```
DROP INDEX idx_department_age ON Student2;
```

3 视图的操作

创建一个数据库 companyinfo，并在该库中创建一个名为"Employee"的表，包括字段(id, name, age, department, salary, hire_date)，并插入一些示例数据，包括但不限于：(1, 'Alice', 28, 'HR', 6500, '2020-01-01'),(2, 'Bob', 32, 'IT', 8000, '2019-05-15'),(3, 'Charlie', 30, 'Sales', 7500, '2021-02-10'),(4, 'David', 31, 'Finance', 7200, '2018-11-20'),(5, 'Emily', 29, 'HR', 5800, '2022-03-05');

（1）创建视图，该视图包含每个部门的总员工人数、平均年龄和平均薪水，名称自拟，需体现出视图的功能性，给出 SQL 语句并输出截图：

```
CREATE DATABASE IF NOT EXISTS companyinfo;
USE companyinfo;
CREATE TABLE IF NOT EXISTS Employee (
    id INT PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    age INT,
    department VARCHAR(50),
    salary DECIMAL(10,2),
    hire_date DATE
);
INSERT INTO Employee (id, name, age, department, salary, hire_date) VALUES
(1, 'Alice', 28, 'HR', 6500.00, '2020-01-01'),
(2, 'Bob', 32, 'IT', 8000.00, '2019-05-15'),
(3, 'Charlie', 30, 'Sales', 7500.00, '2021-02-10'),
(4, 'David', 31, 'Finance', 7200.00, '2018-11-20'),
(5, 'Emily', 29, 'HR', 5800.00, '2022-03-05');
CREATE VIEW DepartmentSummary AS SELECT department, COUNT(name) AS 'number',
AVG(age) AS 'age', AVG(salary) AS 'salary' FROM Employee GROUP BY department;
```

（2）查看该视图的定义，给出 SQL 语句并输出截图：

实 验 报 告

SHOW CREATE VIEW Departmentsummary;

（3）修改视图的定义，仅包含部门为"HR"和"IT"的部门汇总信息，给出 SQL 语句并输出截图：

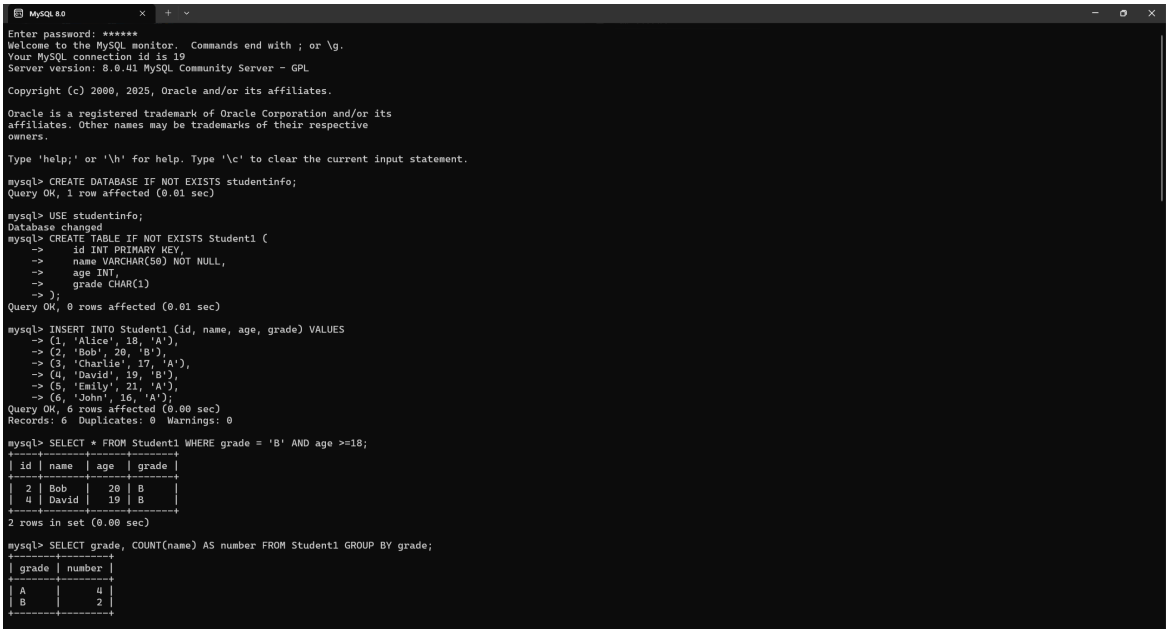
CREATE OR REPLACE VIEW DepartmentSummary AS SELECT department, COUNT(name) AS 'number', AVG(age) AS 'age', AVG(salary) AS 'salary' FROM Employee WHERE department IN ('HR','IT') GROUP BY department;

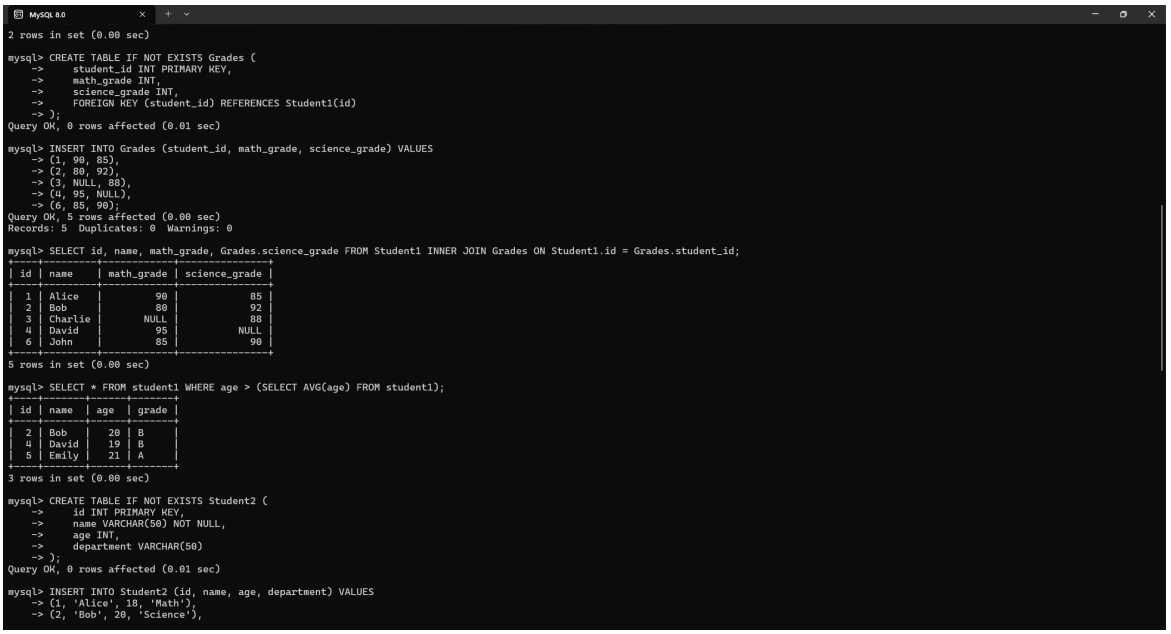
（4）删除视图，给出 SQL 语句并输出截图：

DROP VIEW Departmentsummary;

（5）使用该视图，查询部门为"HR"和"IT"的部门汇总信息，给出 SQL 语句并输出截图：

SELECT * FROM Departmentsummary;





实 验 报 告

```
MySQL 8.0
    -> (3, 'Charlie', 17, 'Math'),
    -> (4, 'David', 19, 'History'),
    -> (5, 'Emily', 21, 'Science');
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> SHOW INDEX FROM Student2 WHERE key_name = 'idx_name';
Empty set (0.00 sec)

mysql> CREATE UNIQUE INDEX idx_department_age ON Student2 (department,age);
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Student2 WHERE department = 'math' AND age = 18;
+----+-------+------+------------+
| id | name  | age  | department |
+----+-------+------+------------+
|  1 | Alice |   18 | Math       |
+----+-------+------+------------+
1 row in set (0.00 sec)

mysql> DROP INDEX idx_department_age ON Student2;
Query OK, 0 rows affected (0.00 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> CREATE DATABASE IF NOT EXISTS companyinfo;
Query OK, 1 row affected (0.00 sec)

mysql> USE companyinfo;
Database changed
mysql> CREATE TABLE IF NOT EXISTS Employee (
    ->     id INT PRIMARY KEY,
    ->     name VARCHAR(50) NOT NULL,
    ->     age INT,
    ->     department VARCHAR(50),
    ->     salary DECIMAL(10,2),
    ->     hire_date DATE
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql> INSERT INTO Employee (id, name, age, department, salary, hire_date) VALUES
    -> (1, 'Alice', 28, 'HR', 6500.00, '2020-01-01'),
    -> (2, 'Bob', 32, 'IT', 8000.00, '2019-05-15'),
    -> (3, 'Charlie', 30, 'Sales', 7500.00, '2021-02-10'),
    -> (4, 'David', 31, 'Finance', 7200.00, '2018-11-20'),
    -> (5, 'Emily', 29, 'HR', 5800.00, '2022-03-05');
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> CREATE VIEW DepartmentSummary AS SELECT department, COUNT(name) AS 'number', AVG(age) AS 'age', AVG(salary) AS 'salary' FROM Employee GROUP BY department;
Query OK, 0 rows affected (0.00 sec)
```

```
MySQL 8.0
Records: 0  Duplicates: 0  Warnings: 0

mysql> CREATE DATABASE IF NOT EXISTS companyinfo;
Query OK, 1 row affected (0.00 sec)

mysql> USE companyinfo;
Database changed
mysql> CREATE TABLE IF NOT EXISTS Employee (
    ->     id INT PRIMARY KEY,
    ->     name VARCHAR(50) NOT NULL,
    ->     age INT,
    ->     department VARCHAR(50),
    ->     salary DECIMAL(10,2),
    ->     hire_date DATE
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql> INSERT INTO Employee (id, name, age, department, salary, hire_date) VALUES
    -> (1, 'Alice', 28, 'HR', 6500.00, '2020-01-01'),
    -> (2, 'Bob', 32, 'IT', 8000.00, '2019-05-15'),
    -> (3, 'Charlie', 30, 'Sales', 7500.00, '2021-02-10'),
    -> (4, 'David', 31, 'Finance', 7200.00, '2018-11-20'),
    -> (5, 'Emily', 29, 'HR', 5800.00, '2022-03-05');
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> CREATE VIEW DepartmentSummary AS SELECT department, COUNT(name) AS 'number', AVG(age) AS 'age', AVG(salary) AS 'salary' FROM Employee GROUP BY department;
Query OK, 0 rows affected (0.00 sec)

mysql> SHOW CREATE VIEW Departmentsummary;
+-------------------+------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------+----------------------+----------------------+
| View              | Create View
                                                                                                  | character_set_client | collation_connection |
+-------------------+------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------+----------------------+----------------------+
| departmentsummary | CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW `departmentsummary` AS select `employee`.`department` AS `department`,count(`employee`.`na
me`) AS `number`,avg(`employee`.`age`) AS `age`,avg(`employee`.`salary`) AS `salary` from `employee` group by `employee`.`department` | gbk                  | gbk_chinese_ci       |
+-------------------+------------------------------------------------------------------------------------------------------------------------
--------------------------------------------------------------------------------------------------+----------------------+----------------------+
1 row in set (0.00 sec)

mysql> CREATE OR REPLACE VIEW DepartmentSummary AS SELECT department, COUNT(name) AS 'number', AVG(age) AS 'age', AVG(salary) AS 'salary' FROM Employee WHERE department IN ('HR','IT') GROUP BY
 department;
Query OK, 0 rows affected (0.00 sec)

mysql> DROP VIEW Departmentsummary;
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT * FROM Departmentsummary;
ERROR 1146 (42S02): Table 'companyinfo.departmentsummary' doesn't exist
mysql> |
```

实 验 报 告

四、实验小结（包括问题和解决方法、心得体会、意见与建议等）

（一） 实验中遇到的主要问题及解决方法

无。

（二）实验心得

本次实验通过实际操作 MySQL 进阶开发，我深入理解了表记录的查询过程，包括单表查询、聚合函数查询、多表连接查询以及子查询的应用，掌握了使用 Navicat for MySQL 进行高效查询的方法。在索引操作部分，我学习了索引的创建、查看、使用和删除，认识到索引在提升查询性能中的重要作用。此外，通过视图的创建、修改和删除，我进一步理解了视图在简化复杂查询和数据安全性方面的优势。实验过程中，我能够顺利完成各项任务，未遇到重大问题，整体操作流畅。通过本次实验，我对 MySQL 的高级功能有了更全面的掌握，为后续数据库开发和应用打下了坚实基础。

（三）意见与建议（没有可省略）

无。

实 验 报 告

五、支撑毕业要求指标点

☑ 4.2-M 能够根据实验方案，配置实验环境、开展实验，综合分析实验结果以获得合理有效的结论。

☑ 5.2-M 能够针对计算机及应用领域中的复杂工程问题，合理选择使用恰当的技术、资源和现代工程工具进行预测和模拟，并理解其局限性。

六、指导教师评语

	评分项	优秀	良好	中等	合格	不合格
评分细则	遵守实验室规章制度					
	学习态度					
	算法思想准备情况					
	程序设计能力					
	解决问题能力					
	算法设计合理性					
	算法效能评价					
	报告书写认真程度					
	内容详实程度					
	文字表达熟练程度					
	其它评价意见					
	本次实验能力达成评价（总成绩）		批阅人		日期	