数据库系统课程实验报告

实验名称:	数据库的完整性
实验日期:	5.9
实验地点:	西部片区 4 号楼 103
提交日期:	5.16

学号:	37220232203786
姓名:	潘腾凯
专业年级:	软工 2023 级
学年学期:	2024-2025 学年第二学期

1. 实验目的

- 理解并掌握关系数据库完整性的运行机制
  - ■ 完整性约束定义>完整性约束检查>违约处理
- 理解并掌握关系数据库完整性主要约束类型及其含义和作用
  - ■ PRIMARY KEY， FOREIGN KEY， NOT NULL， UNIQUE，CHECK
- 理解并掌握关系数据库完整性定义、修改、删除和命名的方法
  - ■ CREATE TABLE， ALTER TABLE
- 熟练掌握 Mysql8.4 下通过系统表查看完整性信息(PK， FK，UNIQUE， CHECK)的方法
  - ■ INFORMATION_SCHEMA.TABLE_CONSTRAINTS 表
  - ■ INFORMATION_SCHEMA.REFERENTIAL_CONSTRAINTS表
  - ■ INFORMATION_SCHEMA.CHECK_CONSTRAINTS 表
  - ■ INFORMATION_SCHEMA.KEY_COLUMN_USAGE 表
  - ■ SHOW CREATE TABLE tbl_name;

2. 实验内容和步骤

本实验分成两部分：一是验证主码、外码、唯一约束和 check 约束的执行效果， 其中包括给约束命名；二是介绍在 mysql8.4 下如何查询已定义的约束信息。

1.约束的定义与效果验证

（1） 创建两张表：雇员表 Emp 和工作表 Work，它们的表结构如下：

Emp 表

字段	含义	数据类型	是否空
Eid	雇员编号	定长字符型，长度为 5	否
Ename	雇员姓名	变长字符型，长度为 10	/
WorkID	工作编号	定长字符，长度为 3	/
Salary	工资	数值型，总长度为 8，包括两位小数	/
Phone	电话号码	定长字符型，长度为 11	否

Work 表

字段	含义	数据类型	是否空
WorkID	工作编号	定长字符，长度为 3	否
LowerSalary	最低工资	数值型，总长度为 8，包括两位小数	/
UpperSalary	最高工资	数值型，总长度为 8，包括两位小数	/

```
mysql> CREATE TABLE Emp (
    Eid CHAR(5) NOT NULL,
    Ename VARCHAR(10),
    WorkID CHAR(3),
    Salary DECIMAL(8, 2),
    Phone CHAR(11) NOT NULL
);
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> CREATE TABLE Work (
    WorkID CHAR(3) NOT NULL,
    LowerSalary DECIMAL(8, 2),
    UpperSalary DECIMAL(8, 2)
);
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> desc Work;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| WorkID      | char(3)     | NO   |     | NULL    |       |
| LowerSalary | decimal(8,2) | YES  |     | NULL    |       |
| UpperSalary | decimal(8,2) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.06 sec)

mysql> desc Emp
    -> ;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| Eid    | char(5)     | NO   |     | NULL    |       |
| Ename  | varchar(10) | YES  |     | NULL    |       |
| WorkID | char(3)     | YES  |     | NULL    |       |
| Salary | decimal(8,2) | YES  |     | NULL    |       |
| Phone  | char(11)    | NO   |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
5 rows in set (0.06 sec)
```

（2） 分别为两张表插入以下数据， 观察插入操作是否成功。

emp 表数据： {(‘10001’,’Smith’,’001’,2000,’13800010001’), (‘10001’, ‘Jonny’, ’001’, 3000, ’13600010002’), (‘10002’, ’Mary’, ’002’, 2500, ’13800020002’)} work 表数据： {(‘001’, 1000, 5000)， (‘002’, 2000, 8000)}

```
mysql> INSERT INTO Emp (Eid, Ename, WorkID, Salary, Phone)
VALUES
('10001', 'Smith', '001', 2000, '13800010001'),
('10001', 'Jonny', '001', 3000, '13600010002'),
('10002', 'Mary', '002', 2500, '13800020002');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> INSERT INTO Work (WorkID, LowerSalary, UpperSalary)
VALUES
('001', 1000, 5000),
('002', 2000, 8000);
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

插入成功

（3） 设置 emp 表的 eid 为主码，观察该操作是否成功。若不成功，

请分析原因并思考如何处理才能添加主码约束成功。 添加主码约束命令： alter table emp add primary key (eid);

```
mysql> ALTER TABLE emp ADD PRIMARY KEY(Eid);
1062 - Duplicate entry '10001' for key 'emp.PRIMARY'
```

添加主码约束失败。

原因：主码是唯一的，前面插入的数据 Eid 有重复项“10001”，所以不能设置为主码约束。

处理：删除重复的数据或修改其中一条数据的 Eid。

先将数据 Eid 修改：

```
mysql> UPDATE emp
SET Eid = '10003'
WHERE Eid = '10001'
  AND Phone = '13600010002'; -- 修改第二条记录
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

再添加主码约束：

```
mysql> ALTER TABLE Emp ADD PRIMARY KEY(Eid);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

这次就成功了。

（4） 在成功设置 eid 为 emp 表的主码后， 发布语句：

alter table emp add constraint pk_emp_eid primary key (eid);

观察执行结果并分析原因， 记住系统的结果提示以便与下面的步骤（10）比较

```
mysql> ALTER TABLE emp
ADD CONSTRAINT pk_emp_eid
PRIMARY KEY(eid);
1068 - Multiple primary key defined
mysql>
```

执行结果分析： 执行上述语句出现该错误提示 ，意思是定义了多个主

键。 原因是之前已经成功将`eid`设置为`emp`表的主键 ，此时再试图将`eid`设置为主键，就相当于为`emp`表定义了多个主键，而在MySQL 中一张表只能有一个主键，所以系统报错。

（5） 分别使用以下两条命令查看主码约束信息，观察结果的异同：

show create table emp;

select * from information_schema.table_constraints where table_name='emp';

```
mysql> SHOW CREATE TABLE Emp;
+-------+-----------------------------------------------------------------
-----------------------------------------------------------------------
-----------------------------------------------------------------------
-----------------------------------------------------+
| Table | Create Table

                                                     |
+-------+-----------------------------------------------------------------
-----------------------------------------------------------------------
-----------------------------------------------------------------------
-----------------------------------------------------+
| Emp   | CREATE TABLE `emp` (
  `Eid` char(5) NOT NULL,
  `Ename` varchar(10) DEFAULT NULL,
  `WorkID` char(3) DEFAULT NULL,
  `Salary` decimal(8,2) DEFAULT NULL,
  `Phone` char(11) NOT NULL,
  PRIMARY KEY (`Eid`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-------+-----------------------------------------------------------------
-----------------------------------------------------------------------
-----------------------------------------------------------------------
-----------------------------------------------------+
1 row in set (0.05 sec)
```

```
mysql> SELECT * FROM information_schema.table_constraints
WHERE table_name='emp';
+--------------------+-------------------+-----------------+--------------+------------+-----------------+----------+
| CONSTRAINT_CATALOG | CONSTRAINT_SCHEMA | CONSTRAINT_NAME | TABLE_SCHEMA | TABLE_NAME | CONSTRAINT_TYPE | ENFORCED |
+--------------------+-------------------+-----------------+--------------+------------+-----------------+----------+
| def                | exp7              | PRIMARY         | exp7         | emp        | PRIMARY KEY     | YES      |
+--------------------+-------------------+-----------------+--------------+------------+-----------------+----------+
1 row in set (0.06 sec)
```

结果相同点：都能确定表存在主键约束。

不同点：

1.信息全面性

第一句：除了主键约束信息外，还展示了整个表的创建语句，包括各字段的定义（如数据类型、是否可为空等）、存储引擎（ENGINE=InnoDB）、字符集（DEFAULT CHARSET=utf8mb4）和排序规则（COLLATE=utf8mb4_0900_ai_ci）等全面的建表信息。

第二句：主要聚焦于表的约束相关信息，只展示了与约束有关的字段，如 CONSTRAINT_CATALOG（约束所属目录）、CONSTRAINT_SCHEMA（约束所属数据库模式）、CONSTRAINT_NAME（约束名称）、TABLE_SCHEMA（表所属数据库模式）、TABLE_NAME（表名）、CONSTRAINT_TYPE（约束类型）、ENFORCED（约束是否生效），信息相对单一但针对性强。

2.信息呈现形式

第一句：以建表语句的形式呈现信息，直观展示表结构的定义方式。

第二句：以查询结果集（表格形式）呈现，每列对应一个约束相关属性，更像是常规的查询结果展示。

（6） 设置 emp 表的 workid 字段为外码，它引用 work 表中的 workid 字段，查看操作是否成功？ 若不成功说明原因， 然后根据系统提示来修改 work 表的结构，使得 work 表满足 emp.workid 为外码的要求

```
mysql> ALTER TABLE emp
ADD FOREIGN KEY (workid) REFERENCES work(workid);
6125 - Failed to add the foreign key constraint. Missing unique key for c
onstraint 'emp_ibfk_1' in the referenced table 'work'
```

操作失败。

原因：MySQL 要求被引用的字段必须是主键或具有唯一约束，而原 work 表的 workid 字段未设置主键或唯一约束。

解决：为 Work 表添加主键约束：

```
mysql> ALTER TABLE work
ADD PRIMARY KEY (workid);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

（7） 第二次执行(6)，设置 emp 表的 workid 为外码，但不给出显式外码约束名，而由系统默认

```
mysql> ALTER TABLE emp
ADD FOREIGN KEY (workid) REFERENCES work(workid);
Query OK, 3 rows affected (0.05 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

（8）再次执行步骤(7)

```
mysql> ALTER TABLE emp
ADD FOREIGN KEY (workid) REFERENCES work(workid);
Query OK, 3 rows affected (0.06 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

（9） 第三次执行步骤(6)， 即设置 emp 表的 workid 为外码， 但要求给出外码约束名，即 fk_emp_work

```
mysql> ALTER TABLE emp
ADD CONSTRAINT fk_emp_work
FOREIGN KEY (workid) REFERENCES work(workid);
Query OK, 3 rows affected (0.05 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

（10） 发布语句： show create table emp； 查看 emp 表的外码约束信息， 理解系统默认的外码约束名的构成及自命名外码约束名， 分

析为什么外码约束名可以有多个，而主码约束只能有一个（对比步骤（4）的结果），总结 mysql 中主码约束命名与外码约束命名的规律

```
mysql> SHOW CREATE TABLE emp;
+-------+------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
| Table | Create Table


+-------+------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
| emp   | CREATE TABLE `emp` (
  `Eid` char(5) NOT NULL,
  `Ename` varchar(10) DEFAULT NULL,
  `WorkID` char(3) DEFAULT NULL,
  `Salary` decimal(8,2) DEFAULT NULL,
  `Phone` char(11) NOT NULL,
  PRIMARY KEY (`Eid`),
  KEY `fk_emp_work` (`WorkID`),
  CONSTRAINT `emp_ibfk_1` FOREIGN KEY (`WorkID`) REFERENCES `work` (`WorkID`),
  CONSTRAINT `emp_ibfk_2` FOREIGN KEY (`WorkID`) REFERENCES `work` (`WorkID`),
  CONSTRAINT `fk_emp_work` FOREIGN KEY (`WorkID`) REFERENCES `work` (`WorkID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-------+------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
------------------------------------------------------------------------------------------------------
1 row in set (0.07 sec)
```

总结

1）系统默认外码约束名构成

以表名开头，接着 ibfk 代表 InnoDB Foreign Key ，后面的数字是序号，按添加外键顺序从 1 开始递增，用于区分同一表不同的外键约束 。

2）自命名外码约束名

如设置的 fk_emp_work，自命名外码约束名可根据业务逻辑和可读性来取，通常采用类似 “fk_主表名_关联字段名” 或 “fk_主表名_从表名” 等形式，方便识别外键关联关系 。

3）外码约束名可多个，主码约束只能一个的原因

外码约束：一个表可以有多个外码，用于关联不同的其他表或同一表不同字段与其他表的关联 。例如 emp 表可能不仅通过 workid 关

联 work 表，还可能通过其他字段关联别的表，所以需要多个外码约束名来区分不同的外键关联关系。

主码约束：一张表只能有一个主键 ，它用于唯一标识表中的每一条记录。如果存在多个主键，就无法明确记录的唯一标识规则，会造成数据逻辑混乱，所以主码约束只能有一个 。

4）MySQL 中主码与外码约束命名规律总结

主码约束命名：一般不特意显式命名时，在 CREATE TABLE 语句中定义主键，系统自动将其关联到字段名；若显式命名，通常采用类似“pk_表名_主键字段名” 形式。

外码约束命名：系统默认命名按 “表名_ibfk_序号” 生成；显式命名建议遵循 “fk_主表名_关联字段名” 或 “fk_主表名_从表名” 等规则 。

（11） 设置 work 表的 workid 为唯一值（唯一约束名由系统默认，无需自命名），连续两次执行该命令

第一次：

```
mysql> ALTER TABLE work ADD UNIQUE (workid);
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

第二次：

```
mysql> ALTER TABLE work ADD UNIQUE (workid);
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 1
```

产生警告。

（12）发布语句： show create table work； 查看 work 表的唯一约束

信息，理解系统默认唯一约束名的构成，总结 mysql 中唯一约束名的规律

```
mysql> SHOW CREATE TABLE work;
+-------+-----------------------------------------------------------------------------
-----------------------------------------------------------------------------------
| Table | Create Table

+-------+-----------------------------------------------------------------------------
-----------------------------------------------------------------------------------
| work  | CREATE TABLE `work` (
  `WorkID` char(3) NOT NULL,
  `LowerSalary` decimal(8,2) DEFAULT NULL,
  `UpperSalary` decimal(8,2) DEFAULT NULL,
  PRIMARY KEY (`WorkID`),
  UNIQUE KEY `WorkID` (`WorkID`),
  UNIQUE KEY `WorkID_2` (`WorkID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-------+-----------------------------------------------------------------------------
-----------------------------------------------------------------------------------
1 row in set (0.06 sec)
```

理解：在同一字段上再次添加唯一约束（虽然逻辑上不合理，但可能因误操作等情况出现），系统会在原字段名基础上添加序号区分，从 2 开始递增。

总结：

未显式命名情况：首次对字段添加唯一约束，系统多以字段名作为约束名；若同一字段重复添加，会在字段名后加序号。

显式命名情况：用户使用 ALTER TABLE 语句添加唯一约束时，可通过 CONSTRAINT 关键字显式命名。

（13） 发布语句： alter table emp add constraint ck_salary check(3000<=salary); 观察操作是否成功？若不成功说明原因

```
mysql> alter table emp add constraint ck_salary check(3000<=salary);
3819 - Check constraint 'ck_salary' is violated.
mysql>
```

操作失败

原因：

emp 表中已存在 salary 值小于 3000 的数据 ，而添加的 CHECK 约束要求 salary 必须大于等于 3000，现有数据与该约束条件冲突，所以系统报错 3819 - Check constraint 'ck_salary' is violated，提示检查约束被违反。

（14） 修改 work 表以保证最低工资 lowersalary 一定不高于最高工资 uppersalary， 要求给出约束名，即 ck_lower_upper_salary

```
mysql> ALTER TABLE work
ADD CONSTRAINT ck_lower_upper_salary CHECK (lowersalary <= uppersalary);
Query OK, 2 rows affected (0.06 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

（15） 给 work 表插入数据(‘003’,4000,3000)，观察操作是否成功? 若不成功说明原因

```
mysql> INSERT INTO Work (WorkID, LowerSalary, UpperSalary)
VALUES
('003', 4000, 3000);
3819 - Check constraint 'ck_lower_upper_salary' is violated.
```

操作不成功。

原因：

在步骤（14）已经添加最低工资要小于等于最高工资的约束，而这条数据的 lowersalary>uppersalary，违反约束，故插入失败。

（16） 发布语句： alter table work add check(lowersalary<=uppersalary); 观察操作是否成功? 若不成功说明原因

```
mysql> ALTER TABLE work ADD CHECK(lowersalary<=uppersalary);
Query OK, 2 rows affected (0.06 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

操作成功

（17）给 work 表插入数据('003',3000,4000)，观察操作是否成功?若不成功说明原因

```
mysql> INSERT INTO Work (WorkID, LowerSalary, UpperSalary)
VALUES
('003', 3000, 4000);
Query OK, 1 row affected (0.01 sec)
```

操作成功

（18）再次发布语句： alter table work add check(lowersalary<=uppersalary); 观察操作是否成功？若不成功说明原因

```
mysql> ALTER TABLE work ADD CHECK(lowersalary<=uppersalary);
Query OK, 3 rows affected (0.06 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

操作成功

（19）发布语句： show create table work； 查看 work 表的 check 约束信息，理解并总结系统默认的 check 约束名的构成方法

```
mysql> SHOW CREATE TABLE work;
+-------+-----------------------------------------------------------------------------
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
| Table | Create Table


+-------+-----------------------------------------------------------------------------
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
| work  | CREATE TABLE `work` (
  `WorkID` char(3) NOT NULL,
  `LowerSalary` decimal(8,2) DEFAULT NULL,
  `UpperSalary` decimal(8,2) DEFAULT NULL,
  PRIMARY KEY (`WorkID`),
  UNIQUE KEY `WorkID` (`WorkID`),
  UNIQUE KEY `WorkID_2` (`WorkID`),
  CONSTRAINT `ck_lower_upper_salary` CHECK ((`LowerSalary` <= `UpperSalary`)),
  CONSTRAINT `work_chk_1` CHECK ((`LowerSalary` <= `UpperSalary`)),
  CONSTRAINT `work_chk_2` CHECK ((`lowersalary` <= `uppersalary`))
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci |
+-------+-----------------------------------------------------------------------------
-------------------------------------------------------------------------------------
-------------------------------------------------------------------------------------
1 row in set (0.07 sec)

mysql>
```

理解与总结：

以表名开头：约束名通常以表名 work 开头，例如 work_chk_1 、work_chk_2 。这里的 work 就是对应表的名称，作为约束名的前缀，方便从约束名上快速识别出该约束属于哪个表。

固定标识：在表名之后，会有固定的标识 chk ，用于表明这是一个 CHECK 约束。这是 MySQL 系统在生成默认 CHECK 约束名时的一种约定，通过这个标识能直观区分约束类型。

序号递增：最后是一个数字序号，从 1 开始递增，如 work_chk_1 、work_chk_2 。当在表上添加多个 CHECK 约束时，系统会按照添加的顺序，依次用递增的数字来区分不同的 CHECK 约束 。

（20） 以下为约束的级联操作验证（这部分在实验 5 中有要求，此

处属于重新复习一遍）。 删除 emp 表上所有的外码约束，重建外码约束， emp 表的 workid 字段引用 work 表的 workid 字段， 要求外码约束中包含 on delete cascade 选项， 外码约束名为 fk_emp_work

删除外码约束：

```
mysql> ALTER TABLE emp
DROP FOREIGN KEY fk_emp_work;
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE emp
DROP FOREIGN KEY emp_ibfk_1;
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> ALTER TABLE emp
DROP FOREIGN KEY emp_ibfk_2;
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

重建：

```
mysql> ALTER TABLE emp
ADD CONSTRAINT fk_emp_work
FOREIGN KEY (workid) REFERENCES work(workid)
ON DELETE CASCADE;
Query OK, 3 rows affected (0.05 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

（21）先查询 emp 表和 work 表中所有数据

```
mysql> SELECT * FROM emp;
+-------+-------+--------+---------+-------------+
| Eid   | Ename | WorkID | Salary  | Phone       |
+-------+-------+--------+---------+-------------+
| 10001 | Smith | 001    | 2000.00 | 13800010001 |
| 10002 | Mary  | 002    | 2500.00 | 13800020002 |
| 10003 | Jonny | 001    | 3000.00 | 13600010002 |
+-------+-------+--------+---------+-------------+
3 rows in set (0.06 sec)

mysql> SELECT * FROM work;
+--------+-------------+-------------+
| WorkID | LowerSalary | UpperSalary |
+--------+-------------+-------------+
| 001    | 1000.00     | 5000.00     |
| 002    | 2000.00     | 8000.00     |
| 003    | 3000.00     | 4000.00     |
+--------+-------------+-------------+
3 rows in set (0.05 sec)
```

（22）发布语句： delete from work where workid='001';

```
mysql> DELETE FROM work
WHERE workid='001';
Query OK, 1 row affected (0.01 sec)
```

（23）发布语句： select * from emp; 观察执行结果并与（21） 中查询 emp 表的结果进行比对，验证 on delete cascade 选项是否发生作用

```
mysql> SELECT * FROM emp;
+-------+-------+--------+---------+-------------+
| Eid   | Ename | WorkID | Salary  | Phone       |
+-------+-------+--------+---------+-------------+
| 10002 | Mary  | 002    | 2500.00 | 13800020002 |
+-------+-------+--------+---------+-------------+
1 row in set (0.04 sec)
```

emp 表中的 Eid 为 10001 和 10003 的记录也一起被删除，说明级联删除选项发挥作用。

2.约束查询

mysql8 中定义的约束信息主要放在 INFORMATION_SCHEMA 数据库的以下表中：

■ TABLE_CONSTRAINTS 表（主码、外码、唯一约束、 check 约束均可查询该表得到，如果存在）

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE	ENFORCED
def	exp7	PRIMARY	exp7	work	PRIMARY KEY	YES
def	exp7	WorkID	exp7	work	UNIQUE	YES
def	exp7	WorkID_2	exp7	work	UNIQUE	YES
def	exp7	ck_lower_upper_salary	exp7	work	CHECK	YES
def	exp7	work_chk_1	exp7	work	CHECK	YES
def	exp7	work_chk_2	exp7	work	CHECK	YES
def	exp7	PRIMARY	exp7	emp	PRIMARY KEY	YES
def	exp7	fk_emp_work	exp7	emp	FOREIGN KEY	YES

174 rows in set (0.17 sec)

■ REFERENTIAL_CONSTRAINTS 表（提供外码引用的信息，包括删除规则，更新规则）

```
mysql> select * from REFERENTIAL_CONSTRAINTS;
```

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	UNIQUE_CONSTRAINT_CATALOG	UNIQUE_CONSTRAINT_SCHEMA	UNIQUE_CONSTRAINT_NAME	MATCH_OPTION	UPDATE_RULE	DELETE_RULE	TABLE_NAME	REFERENCED_TABLE_NAME
def	exp4	employees_ibfk_1	def	exp4	PRIMARY	NONE	CASCADE	NO ACTION	employees	departments
def	sales	fk_contacts_customer	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	contacts	customers
def	sales	fk_countries_region	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	countries	regions
def	sales	fk_employees_manager	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	employees	employees
def	sales	fk_inventories_product	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	inventories	products
def	sales	fk_inventories_warehouse	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	inventories	warehouses
def	sales	fk_locations_country	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	locations	countries
def	sales	fk_order_items_order	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	order_items	orders
def	sales	fk_order_items_product	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	order_items	products
def	sales	fk_orders_customer	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	orders	customers
def	sales	fk_orders_salesman	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	orders	employees
def	sales	fk_products_category	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	products	product_categories
def	sales	fk_warehouses_location	def	sales	PRIMARY	NONE	RESTRICT	RESTRICT	warehouses	locations
def	exp7	fk_emp_work	def	exp7	PRIMARY	NONE	NO ACTION	CASCADE	emp	work

■ CHECK_CONSTRAINTS 表（提供 check 所在的库、 check 约束名及 check 语句内容等信息）

```
mysql> select * from CHECK_CONSTRAINTS;
+--------------------+-------------------+-----------------------+----------------------------------+
| CONSTRAINT_CATALOG | CONSTRAINT_SCHEMA | CONSTRAINT_NAME       | CHECK_CLAUSE                     |
+--------------------+-------------------+-----------------------+----------------------------------+
| def                | exp7              | ck_lower_upper_salary | (`LowerSalary` <= `UpperSalary`) |
| def                | exp7              | work_chk_1            | (`LowerSalary` <= `UpperSalary`) |
| def                | exp7              | work_chk_2            | (`lowersalary` <= `uppersalary`) |
+--------------------+-------------------+-----------------------+----------------------------------+
```

■ KEY_COLUMN_USAGE 表（提供键列上约束的详细信息，包括主码、外码）

```
mysql> select * from KEY_COLUMN_USAGE WHERE CONSTRAINT_SCHEMA = 'exp7';
+--------------------+--------------------+-----------------+--------------+--------------+------------+-------------+------------------+------------------------------+------------------------+-----------------------+-------------------------+
| CONSTRAINT_CATALOG | CONSTRAINT_SCHEMA  | CONSTRAINT_NAME | TABLE_CATALOG | TABLE_SCHEMA | TABLE_NAME | COLUMN_NAME | ORDINAL_POSITION | POSITION_IN_UNIQUE_CONSTRAINT | REFERENCED_TABLE_SCHEMA | REFERENCED_TABLE_NAME | REFERENCED_COLUMN_NAME |
+--------------------+--------------------+-----------------+--------------+--------------+------------+-------------+------------------+------------------------------+------------------------+-----------------------+-------------------------+
| def                | exp7               | PRIMARY         | def          | exp7         | emp        | Eid         |                1 | NULL                         | NULL                   | NULL                  | NULL                    |
| def                | exp7               | fk_emp_work     | def          | exp7         | emp        | WorkID      |                1 |                            1 | exp7                   | work                  | WorkID                  |
| def                | exp7               | PRIMARY         | def          | exp7         | work       | WorkID      |                1 | NULL                         | NULL                   | NULL                  | NULL                    |
| def                | exp7               | WorkID          | def          | exp7         | work       | WorkID      |                1 | NULL                         | NULL                   | NULL                  | NULL                    |
| def                | exp7               | WorkID_2        | def          | exp7         | work       | WorkID      |                1 | NULL                         | NULL                   | NULL                  | NULL                    |
+--------------------+--------------------+-----------------+--------------+--------------+------------+-------------+------------------+------------------------------+------------------------+-----------------------+-------------------------+
5 rows in set (0.09 sec)
```

以上表结构都可以使用 desc tbl_name;命令得到，如 desc table_constraints;

如果要获得某张表上约束的定义语句，可使用以下命令获得：

■ SHOW CREATE TABLE tbl_name; -- 将 tbl_name 替换为实际的表

查询上述 4 张表并验证表中信息是否为第一部分(1.约束的定义与效果验证）操作中的主码、外码、唯一约束和 check 约束信息

观察以上结果，经验证与第一部分约束信息一致。

3.删除表上的所有约束

（1）删除 emp 表上的所有约束(包括主码、外码、唯一约束和 check 约束)

（2）删除 work 表上的所有约束(包括主码、外码、唯一约束和 check 约束)

3. 实验总结

3.1 完成的工作

主码、外码、唯一约束和 check 约束的定义与删除，约束的查询。

3.2 对实验的认识

思考题：

1.总结 mysql8.4 中以下约束名的规律： 主码、外码、 唯一约束和 check 约束

答：MySQL 中主码与外码约束命名规律总结（详见步骤 10）

主码约束命名：一般不特意显式命名时，在 CREATE TABLE 语句中定义主键，系统自动将其关联到字段名；若显式命名，通常采用类似 “pk_表名_主键字段名” 形式。

外码约束命名：系统默认命名按 “表名_ibfk_序号” 生成；显式命名建议遵循 “fk_主表名_关联字段名” 或 “fk_主表名_从表名” 等规则 。

唯一约束名规律（详见 12）：未显式命名情况：首次对字段添加唯一约束，系统多以字段名作为约束名；若同一字段重复添加，会在字段名后加序号。显式命名情况：用户使用 ALTER TABLE 语句添加唯一约束时，可通过 CONSTRAINT 关键字显式命名。

CHECK（详见步骤 19）：默认的 CHECK 约束名构成形式为 “表名_chk_序号”（按序号递增）。

2.对于一个相同的约束，如 alter table work unique(workid);可能有多个

不同的约束名，请问这些不同的约束名对应不同的约束效果还是同一个约束效果？ 即， 具有不同约束名的同一约束的效果是否不同？

答：对于相同的约束（如 UNIQUE 约束 ），多个不同的约束名对应的是同一个约束效果。以 alter table work unique(workid);为例，不管是系统默认生成的约束名，还是显式指定的约束名，其作用都是确保 work 表的 workid 字段值具有唯一性 。

3.3 遇到的困难及解决方法

无。