数据库系统课程实验报告

批注 [m1]: 文件名：学号-姓名-实验#，如 00000-张三-实验 2

实验名称:	**数据更新**
实验日期:	2025/4/25
实验地点:	文宣楼 B308
提交日期:	4/25

学号:	36720232204041
姓名:	苏一涵
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

- 熟练掌握单条记录和小批量数据插入的方法（INSERT）
- 熟练掌握使用子查询实现数据插入的方法(INSERT INTO...SUBQUERY)
- 熟练掌握数据修改和删除的方法（UPDATE, DELETE, TRUNCATE）
- 理解约束对数据更新的影响

2.实验内容和步骤

该部分的写作格式示例如下：

（1）为地区表 regions 新增一条记录：（‘5’，‘Oceania’）

步骤如下：

直接插入即可

```
mysql> INSERT INTO regions (region_id, region_name)
VALUES ('5', 'Oceania');
Query OK, 1 row affected (0.01 sec)
```

region_id # int	region_name ABC varchar(50)
1	Europe
2	Americas
3	Asia
4	Middle East and Africa
5	Oceania

（2）将 countries 表中的国家名为 Australia 的 region_id 改为 5

步骤如下：

```
mysql> UPDATE countries
SET region_id = 5
WHERE country_name = 'Australia';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql>
```

country_id ABC char(2)	country_name ABC varchar(40)	region_id # int
AR	Argentina	2
AU	Australia	5
BE	Belgium	1
BR	Brazil	2

（3）使用一条批量插入数据语句为 countries 表新增 5 条记录：('NO','Norway','1'), ('ES','Spain','1'),('SE','Sweden','1'), ('PT','Portugal','1'), ('NZ','New Zealand','5')

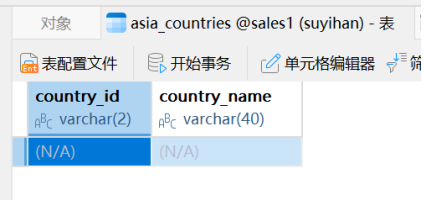
步骤如下：直接插入即可

```
mysql> INSERT INTO countries (country_id, country_name, region_id)
VALUES ('NO','Norway','1'), ('ES','Spain','1'),('SE','Sweden','1'), ('PT','Portugal','1'), ('NZ','New Zealand','5'
);
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

（4）创建一张名为 Asia_countries(country_id, country_name)的新表，其中字段就是 countries 表中的同名字段

步骤如下：

```
mysql> CREATE TABLE Asia_countries (
    country_id VARCHAR(2),
    country_name VARCHAR(40)
);
Query OK, 0 rows affected (0.02 sec)
```



（5）将 countries 表中所有亚洲国家的数据插入到该表中（要求使用插入子查询结果的方法实现）

步骤如下：

将 regin_id=3 的都插入 asia_country 即可

```
mysql> INSERT INTO Asia_countries (country_id, country_name)
SELECT country_id, country_name
FROM countries
WHERE region_id = 3;
Query OK, 5 rows affected (0.00 sec)
Records: 5  Duplicates: 0  Warnings: 0
```

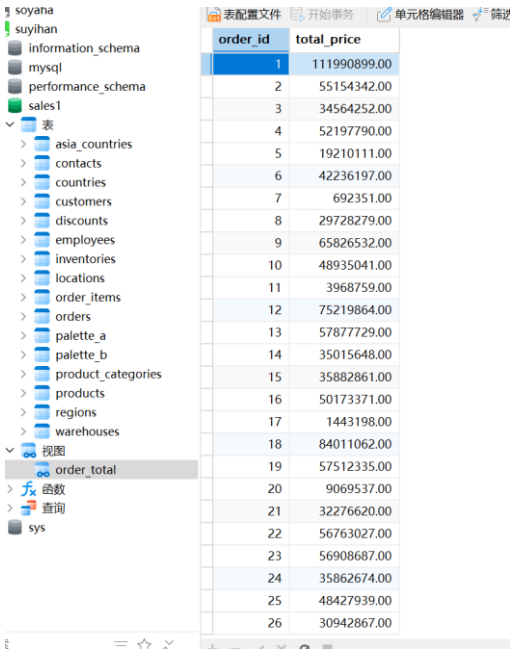
表配置文件 开始事务 单元格编辑器

country_id varchar(2)	country_name varchar(40)
CN	China
IN	India
JP	Japan
ML	Malaysia
SG	Singapore

（6）创建一张名为 order_total(order_id, total_price)的视图，该视图存放每个订单号及其总价，其中 total_price 为总价，其值为数量 quantity 与单价 unit_price 乘积之和， order_id， quantity 和 unit_price 为 order_items 表中的同名字段

步骤如下：首先，使用 CREATE VIEW 语句，指定视图名称和字段 order_id 和 total_price。然后，子查询部分需要从 order_items 表中选择，按 order_id 分组，使用 SUM 函数计算每个订单的总和，也就是 SUM (quantity * unit_price)。需要确保分组正确，每个订单号对应一个总价

```
mysql> CREATE VIEW order_total AS
SELECT
    order_id,
    SUM(quantity * unit_price) AS total_price
FROM
    order_items
GROUP BY
    order_id;
Query OK, 0 rows affected (0.01 sec)
```



order_id	total_price
1	111990899.00
2	55154342.00
3	34564252.00
4	52197790.00
5	19210111.00
6	42236197.00
7	692351.00
8	29728279.00
9	65826532.00
10	48935041.00
11	3968759.00
12	75219864.00
13	57877729.00
14	35015648.00
15	35882861.00
16	50173371.00
17	1443198.00
18	84011062.00
19	57512335.00
20	9069537.00
21	32276620.00
22	56763027.00
23	56908687.00
24	35862674.00
25	48427939.00
26	30942867.00

（7）查询 order_total 视图中订单号 order_id 为 97 的总价并记录该结果

步骤如下：

```
mysql> SELECT total_price
FROM order_total
WHERE order_id = 97;
+-------------+
| total_price |
+-------------+
| 61676319.00 |
+-------------+
1 row in set (0.03 sec)
```

（8）将 order_items 表中 product_id 为 99 的单价 unit_price 增加 4 元

步骤如下：SET unit_price = unit_price + 4：将 unit_price 字段的值增加

4 元。

WHERE product_id = 99：筛选条件，确保只更新 product_id 为 99 的记录

```
mysql> UPDATE order_items
SET unit_price = unit_price + 4
WHERE product_id = 99;
Query OK, 2 rows affected (0.01 sec)
Rows matched: 2  Changed: 2  Warnings: 0
```

（9）查询视图 order_total 中订单号 order_id 为 97 的总价，将其与第（7）步结果进行比较，观察其异同

步骤如下：

```
mysql> SELECT total_price
FROM order_total
WHERE order_id = 97;
+-------------+
| total_price |
+-------------+
| 61676511.00 |
+-------------+
1 row in set (0.03 sec)
```

与第七步对比可以发现总价增加了 4x 倍，x 为总数目

（10）使用 delete 命令删除 Asia_countries 表中 country_id 为 IN 的记录

步骤如下：

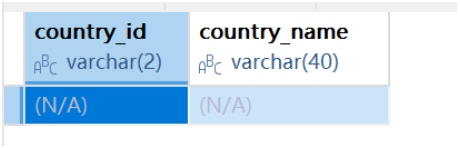
```
mysql> DELETE FROM Asia_countries
WHERE country_id = 'IN';
Query OK, 1 row affected (0.00 sec)
```

country_id varchar(2)	country_name varchar(40)
CN	China
JP	Japan
ML	Malaysia
SG	Singapore

（11）使用 truncate 命令清空 Asia_countries 表的所有记录

步骤如下：

```
mysql> TRUNCATE TABLE Asia_countries;
Query OK, 0 rows affected (0.02 sec)
```



country_id	country_name
varchar(2)	varchar(40)
(N/A)	(N/A)

（12）删除 Asia_countries 表和视图 order_total

步骤如下：用 drop 直接删除

```
mysql> DROP TABLE IF EXISTS Asia_countries;
Query OK, 0 rows affected (0.01 sec)

mysql> DROP VIEW IF EXISTS order_total;
Query OK, 0 rows affected (0.01 sec)
```

（13）使用 show create table employees; 命令查看 employees 表的外键约束。

若结果显示没有外键约束，则使用以下命令建立外键约束：

alter table employees add constraint fk_employees_manager foreign key (manager_id) references employees(employee_id);

若有外键约束且出现 ON DELETE CASCADE 选项（下图），则先删后建外键约束，保证不出现 ON DELETE CASCADE 选项

```
KEY `fk_employees_manager` (`manager_id`),
CONSTRAINT `fk_employees_manager` FOREIGN KEY (`manager_id`) REFERENCES `employees` (`employee_id`) ON DELETE CASCADE
```

若有外键约束但没有 ON DELETE CASCADE 选项，则进行第（14）步。

步骤如下：

```
mysql> show create table employees;
| Table     | Create Table |
| employees | CREATE TABLE `employees` (
  `employee_id` int NOT NULL,
  `first_name` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `last_name` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `email` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `phone` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT NULL,
  `hire_date` date NOT NULL,
  `manager_id` int DEFAULT NULL,
  `job_title` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT NULL,
  PRIMARY KEY (`employee_id`) USING BTREE,
  UNIQUE KEY `email` (`email`) USING BTREE,
  KEY `fk_employees_manager` (`manager_id`) USING BTREE,
  CONSTRAINT `employees_ibfk_1` FOREIGN KEY (`manager_id`) REFERENCES `employees` (`employee_id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  CONSTRAINT `fk_employees_manager` FOREIGN KEY (`manager_id`) REFERENCES `employees` (`employee_id`) ON DELETE RESTRICT ON UPDATE RESTRICT
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci ROW_FORMAT=DYNAMIC |
1 row in set (0.04 sec)
```

由图可知 employees 表存在外键约束。有两个外键约束，分别是 employees_ibfk_1 和 fk_employees_manager，它们均定义了 manager_id 字段参照 employees 表的 employee_id 字段，并且都带有 ON DELETE RESTRICT 和 ON UPDATE RESTRICT 选项，这意味着在删除或更新父表记录时，如果子表中有相关记录，操作会受到限制。

由于没有 ON DELETE CASCADE 选项，所以无需对现有外键进行修改

（14）查询 employees 表中 manager_id 为 1 的记录

步骤如下：

```
mysql> SELECT *
FROM employees
WHERE manager_id = 1;
```

employee_id	first_name	last_name	email	phone	hire_date	manager_id	job_title
2	Jude	Rivera	juderivera@examplecom	5151234568	2021-09-16	1	Administration Vice President
3	Blake	Cooper	blakecooper@examplecom	5151234569	2013-09-16	1	Administration Vice President
15	Rory	Kelly	rorykelly@examplecom	5151274561	2007-12-16	1	Purchasing Manager
21	Jaxon	Ross	jaxonross@examplecom	6501231234	2018-07-16	1	Stock Manager
22	Liam	Henderson	liamhenderson@examplecom	6501232234	2010-02-16	1	Stock Manager
23	Jackson	Coleman	jacksoncoleman@examplecom	6501233234	2001-05-16	1	Stock Manager
24	Callum	Jenkins	callumjenkins@examplecom	6501234234	2010-10-16	1	Stock Manager
25	Ronnie	Perry	ronnieperry@examplecom	6501235234	2016-11-16	1	Stock Manager
46	Ava	Sullivan	avasullivan@examplecom	011441344429268	2001-10-16	1	Sales Manager
47	Ella	Wallace	ellawallace@examplecom	011441344467268	2005-09-16	1	Sales Manager
48	Jessica	Woods	jessicawoods@examplecom	011441344429278	2010-03-16	1	Sales Manager
49	Isabella	Cole	isabellacole@examplecom	011441344619268	2015-10-16	1	Sales Manager
50	Mia	West	miawest@examplecom	011441344429018	2029-09-16	1	Sales Manager
102	Emma	Perkins	emmaperkins@examplecom	5151235555	2017-02-16	1	Marketing Manager

14 rows in set (0.05 sec)

（15）执行以下删除命令：

DELETE FROM employees where manager_id=1;

观察执行结果

步骤如下：

结果如下

```
mysql> DELETE FROM employees where manager_id=1;
1451 - Cannot delete or update a parent row: a foreign key constraint fails (`sales1`.`employees`, CONSTRAINT `employees_ibfk_1` FOREIGN KEY
(`manager_id`) REFERENCES `employees` (`employee_id`) ON DELETE RESTRICT ON UPDATE RESTRICT)
```

错误表明，由于 employees 表存在外键约束 employees_ibfk_1，导致无法直接删除 manager_id = 1 的记录。

（16）删除 employees 表上的外键约束 fk_employees_manager，然后使用以下命令重建该外键约束：

alter table employees add constraint fk_employees_manager FOREIGN KEY(manager_id) REFERENCES employees(employee_id) ON DELETE CASCADE;

执行后使用命令 show create table employees;查看结果

```
KEY `fk_employees_manager` (`manager_id`),
CONSTRAINT `fk_employees_manager` FOREIGN KEY (`manager_id`) REFERENCES `employees` (`employee_id`) ON DELETE CASCADE
```

这一步的目的是测试 ON DELETE CASCADE 的作用和效果

步骤如下：

删除外键约束，并重建外键约束

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

mysql> alter table employees add constraint fk_employees_manager FOREIGN KEY(manager_id)
REFERENCES employees(employee_id) ON DELETE CASCADE;
Query OK, 111 rows affected (0.03 sec)
Records: 111  Duplicates: 0  Warnings: 0
```

查看结果如下

```
mysql> show create table employees;
+-----------+-------------------------------+
| Table     | Create Table                  |
+-----------+-------------------------------+
| employees | CREATE TABLE `employees` (
  `employee_id` int NOT NULL,
  `first_name` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `last_name` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `email` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci NOT NULL,
  `phone` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT NULL,
  `hire_date` date NOT NULL,
  `manager_id` int DEFAULT NULL,
  `job_title` varchar(255) CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci DEFAULT NULL,
  PRIMARY KEY (`employee_id`) USING BTREE,
  UNIQUE KEY `email` (`email`) USING BTREE,
  KEY `fk_employees_manager` (`manager_id`) USING BTREE,
  CONSTRAINT `employees_ibfk_1` FOREIGN KEY (`manager_id`) REFERENCES `employees` (`employee_id`) ON DELETE RESTRICT ON UPDATE RESTRICT,
  CONSTRAINT `fk_employees_manager` FOREIGN KEY (`manager_id`) REFERENCES `employees` (`employee_id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci ROW_FORMAT=DYNAMIC |
+-----------+-------------------------------+
1 row in set (0.05 sec)
```

（17）执行删除命令 DELETE FROM employees where manager_id=1;

步骤如下：

```
mysql> DELETE FROM employees where manager_id=1;
1451 - Cannot delete or update a parent row: a foreign key constraint fails (`sales1`.`employees`, CONSTRAINT `employees_ibfk_1` FOREIGN KEY (`manager_id`) REFERENCES `employees` (`employee_id`) ON DELETE RESTRICT ON UPDATE RESTRICT)
```

执行失败，因为有外键约束。

（18）查询 employees 表中 manager_id 为 1 的记录，观察执行结果

步骤如下：

```
mysql> SELECT *
FROM employees
WHERE manager_id = 1;
```

employee_id	first_name	last_name	email	phone	hire_date	manager_id	job_title
2	Jude	Rivera	juderivera@examplecom	5151234568	2021-09-16	1	Administration Vice President
3	Blake	Cooper	blakecooper@examplecom	5151234569	2013-09-16	1	Administration Vice President
15	Rory	Kelly	rorykelly@examplecom	5151274561	2007-12-16	1	Purchasing Manager
21	Jaxon	Ross	jaxonross@examplecom	6501231234	2018-07-16	1	Stock Manager
22	Liam	Henderson	liamhenderson@examplecom	6501232234	2010-02-16	1	Stock Manager
23	Jackson	Coleman	jacksoncoleman@examplecom	6501233234	2001-05-16	1	Stock Manager
24	Callum	Jenkins	callumjenkins@examplecom	6501234234	2010-10-16	1	Stock Manager
25	Ronnie	Perry	ronnieperry@examplecom	6501235234	2016-11-16	1	Stock Manager
46	Ava	Sullivan	avasullivan@examplecom	011441344429268	2001-10-16	1	Sales Manager
47	Ella	Wallace	ellawallace@examplecom	011441344467268	2005-09-16	1	Sales Manager
48	Jessica	Woods	jessicawoods@examplecom	011441344429278	2010-03-16	1	Sales Manager
49	Isabella	Cole	isabellacole@examplecom	011441344619268	2015-10-16	1	Sales Manager
50	Mia	West	miawest@examplecom	011441344429018	2029-09-16	1	Sales Manager
102	Emma	Perkins	emmaperkins@examplecom	5151235555	2017-02-16	1	Marketing Manager

14 rows in set (0.05 sec)

（19）查询 employees 表中的所有记录，观察执行结果

步骤如下：

```
mysql> SELECT * FROM employees;
```

employee_id	first_name	last_name	email	phone	hire_date	manager_id	job_title
1	Tommy	Bailey	tommybailey@examplecom	5151234567	2017-06-16	NULL	President
2	Jude	Rivera	juderivera@examplecom	5151234568	2021-09-16	1	Administration Vice President
3	Blake	Cooper	blakecooper@examplecom	5151234569	2013-09-16	1	Administration Vice President
4	Louie	Richardson	louierichardson@examplecom	5904234567	2003-09-16	3	Programmer
5	Nathan	Cox	nathancox@examplecom	5904234568	2021-05-16	4	Programmer
6	Gabriel	Howard	gabrielhoward@examplecom	5904234569	2025-06-16	4	Programmer
7	Charles	Ward	charlesward@examplecom	5904234560	2005-02-16	4	Programmer
8	Bobby	Torres	bobbytorres@examplecom	5904235567	2007-02-16	4	Programmer
9	Mohammad	Peterson	mohammadpeterson@examplecom	5151244569	2017-09-16	2	Finance Manager
10	Ryan	Gray	ryangray@examplecom	5151244169	2016-09-16	9	Accountant
11	Tyler	Ramirez	tylerramirez@examplecom	5151244269	2028-09-16	9	Accountant
12	Elliott	James	elliottjames@examplecom	5151244369	2030-09-16	9	Accountant
13	Albert	Watson	albertwatson@examplecom	5151244469	2007-03-16	9	Accountant
14	Elliot	Brooks	elliotbrooks@examplecom	5151244567	2007-12-16	9	Accountant
15	Rory	Kelly	rorykelly@examplecom	5151274561	2007-12-16	1	Purchasing Manager
16	Alex	Sanders	alexsanders@examplecom	5151274562	2018-05-16	15	Purchasing Clerk
17	Frederick	Price	frederickprice@examplecom	5151274563	2024-12-16	15	Purchasing Clerk
18	Ollie	Bennett	olliebennett@examplecom	5151274564	2024-07-16	15	Purchasing Clerk
19	Louis	Wood	louiswood@examplecom	5151274565	2015-11-16	15	Purchasing Clerk
20	Dexter	Barnes	dexterbarnes@examplecom	5151274566	2010-09-16	15	Purchasing Clerk
21	Jaxon	Ross	jaxonross@examplecom	6501231234	2018-07-16	1	Stock Manager
22	Liam	Henderson	liamhenderson@examplecom	6501231334	2010-02-16	1	Stock Manager
23	Jackson	Coleman	jacksoncoleman@examplecom	6501233234	2001-05-16	1	Stock Manager
24	Callum	Jenkins	callumjenkins@examplecom	6501234234	2010-10-16	1	Stock Manager
25	Ronnie	Perry	ronnieperry@examplecom	6501235234	2016-11-16	1	Stock Manager
26	Leon	Powell	leonpowell@examplecom	6501241214	2016-07-16	21	Stock Clerk
27	Kai	Long	kailong@examplecom	6501241224	2028-09-16	21	Stock Clerk
28	Aaron	Patterson	aaronpatterson@examplecom	6501241334	2014-09-16	21	Stock Clerk
29	Roman	Hughes	romanhughes@examplecom	6501241434	2008-03-16	21	Stock Clerk
30	Austin	Flores	austinflores@examplecom	6501245234	2020-09-16	22	Stock Clerk
31	Ellis	Washington	elliswashington@examplecom	6501246234	2030-10-16	22	Stock Clerk
32	Jamie	Butler	jamiebutler@examplecom	6501247234	2016-02-16	22	Stock Clerk
33	Reggie	Simmons	reggiesimmons@examplecom	6501248234	2010-02-16	22	Stock Clerk
34	Seth	Foster	sethfoster@examplecom	6501271934	2014-06-16	23	Stock Clerk
35	Carter	Gonzales	cartergonzales@examplecom	6501271834	2026-09-16	23	Stock Clerk
36	Felix	Bryant	felixbryant@examplecom	6501271734	2012-12-16	23	Stock Clerk
37	Ibrahim	Alexander	ibrahimalexander@examplecom	6501271634	2006-02-16	23	Stock Clerk
38	Sonny	Russell	sonnyrussell@examplecom	6501211234	2014-07-16	24	Stock Clerk
39	Kian	Griffin	kiangriffin@examplecom	6501212034	2026-10-16	24	Stock Clerk
40	Caleb	Diaz	calebdiaz@examplecom	6501212019	2012-02-16	24	Stock Clerk
41	Connor	Hayes	connorhayes@examplecom	6501211834	2006-02-16	24	Stock Clerk
42	Amelia	Myers	ameliamyers@examplecom	6501218009	2017-10-16	25	Stock Clerk
43	Olivia	Ford	oliviaford@examplecom	6501212994	2029-09-16	25	Stock Clerk
44	Emily	Hamilton	emilyhamilton@examplecom	6501212874	2015-03-16	25	Stock Clerk
45	Isla	Graham	islagraham@examplecom	6501212004	2009-07-16	25	Stock Clerk
46	Ava	Sullivan	avasullivan@examplecom	011441344429268	2001-10-16	1	Sales Manager
47	Ella	Wallace	ellawallace@examplecom	011441344467268	2005-09-16	1	Sales Manager
48	Jessica	Woods	jessicawoods@examplecom	011441344429278	2010-03-16	1	Sales Manager
49	Isabella	Cole	isabellacole@examplecom	011441344619268	2015-10-16	1	Sales Manager
50	Mia	West	miawest@examplecom	011441344429018	2029-09-16	1	Sales Manager
51	Poppy	Jordan	poppyjordan@examplecom	011441344129268	2030-09-16	46	Sales Representative
52	Sophie	Owens	sophieowens@examplecom	011441344345268	2024-03-16	46	Sales Representative
53	Sophia	Reynolds	sophiareynolds@examplecom	011441344478968	2020-09-16	46	Sales Representative
54	Lily	Fisher	lilyfisher@examplecom	011441344498718	2030-03-16	46	Sales Representative
55	Grace	Ellis	graceellis@examplecom	011441344987668	2009-12-16	46	Sales Representative
56	Evie	Harrison	evieharrison@examplecom	011441344486508	2023-11-16	46	Sales Representative
57	Scarlett	Gibson	scarlettgibson@examplecom	011441345429268	2030-09-16	47	Sales Representative
58	Ruby	Mcdonald	rubymcdonald@examplecom	011441345929268	2004-03-16	47	Sales Representative
59	Chloe	Cruz	chloecruz@examplecom	011441345829268	2001-09-16	47	Sales Representative
60	Isabelle	Marshall	isabellemarshall@examplecom	011441345729268	2010-03-16	47	Sales Representative
61	Daisy	Ortiz	daisyortiz@examplecom	011441345629268	2015-12-16	47	Sales Representative
62	Freya	Gomez	freyagomez@examplecom	011441345529268	2003-11-16	47	Sales Representative
63	Phoebe	Murray	phoebemurray@examplecom	011441346129268	2011-11-16	48	Sales Representative
64	Florence	Freeman	florencefreeman@examplecom	011441346229268	2019-03-16	48	Sales Representative
65	Alice	Wells	alicewells@examplecom	011441346329268	2024-09-16	48	Sales Representative
66	Charlotte	Webb	charlottewebb@examplecom	011441346529268	2023-02-16	48	Sales Representative
67	Sienna	Simpson	siennasimpson@examplecom	011441346629268	2024-03-16	48	Sales Representative
68	Matilda	Stevens	matildastevens@examplecom	011441346729268	2021-02-16	48	Sales Representative
69	Evelyn	Tucker	evelyntucker@examplecom	011441343929268	2011-03-16	49	Sales Representative
70	Eva	Porter	evaporter@examplecom	011441343829268	2023-03-16	49	Sales Representative
71	Millie	Hunter	milliehunter@examplecom	011441343729268	2024-09-16	49	Sales Representative
72	Sofia	Hicks	sofiahicks@examplecom	011441343629268	2021-02-16	49	Sales Representative
73	Lucy	Crawford	lucycrawford@examplecom	011441343529268	2024-03-16	49	Sales Representative
74	Elsie	Henry	elsiehenry@examplecom	011441343329268	2021-02-16	49	Sales Representative
75	Imogen	Boyd	imogenboyd@examplecom	011441644429267	2011-05-16	50	Sales Representative
76	Layla	Mason	laylamason@examplecom	011441644429266	2019-03-16	50	Sales Representative
77	Rosie	Morales	rosiemorales@examplecom	011441644429265	2024-03-16	50	Sales Representative
78	Maya	Kennedy	mayakennedy@examplecom	011441644429264	2023-02-16	50	Sales Representative
79	Esme	Warren	esmewarren@examplecom	011441644429263	2024-05-16	50	Sales Representative
80	Elizabeth	Dixon	elizabethdixon@examplecom	011441644429262	2004-09-16	50	Sales Representative
81	Lola	Ramos	lolaramos@examplecom	6505079876	2024-09-16	21	Shipping Clerk
82	Willow	Reyes	willowreyes@examplecom	6505079877	2023-02-16	21	Shipping Clerk
83	Ivy	Burns	ivyburns@examplecom	6505079878	2021-06-16	21	Shipping Clerk
84	Erin	Gordon	eringordon@examplecom	6505079879	2003-02-16	21	Shipping Clerk
85	Holly	Shaw	hollyshaw@examplecom	6505091876	2027-09-16	22	Shipping Clerk
86	Emilia	Holmes	emiliaholmes@examplecom	6505092876	2020-02-16	22	Shipping Clerk
87	Molly	Rice	mollyrice@examplecom	6505093876	2024-06-16	22	Shipping Clerk
88	Ellie	Robertson	ellierobertson@examplecom	6505094876	2007-02-16	22	Shipping Clerk
89	Jasmine	Hunt	jasminehunt@examplecom	6505051876	2014-06-16	23	Shipping Clerk
90	Eliza	Black	elizablack@examplecom	6505052876	2013-09-16	23	Shipping Clerk
91	Lilly	Daniels	lillydaniels@examplecom	6505053876	2011-07-16	23	Shipping Clerk
92	Abigail	Palmer	abigailpalmer@examplecom	6505054876	2019-12-16	23	Shipping Clerk
93	Georgia	Mills	georgiamills@examplecom	6505011876	2004-02-16	24	Shipping Clerk
94	Maisie	Nichols	maisienichols@examplecom	6505212876	2003-03-16	24	Shipping Clerk
95	Eleanor	Grant	eleanorgrant@examplecom	6505013876	2001-07-16	24	Shipping Clerk
96	Hannah	Knight	hannahknight@examplecom	6505014876	2017-03-16	24	Shipping Clerk
97	Harriet	Ferguson	harrietferguson@examplecom	6505079811	2024-02-16	25	Shipping Clerk
98	Amber	Rose	amberrose@examplecom	6505079822	2023-05-16	25	Shipping Clerk
99	Bella	Stone	bellastone@examplecom	6505079833	2021-06-16	25	Shipping Clerk
100	Thea	Hawkins	theahawkins@examplecom	6505079844	2013-09-16	25	Shipping Clerk
101	Annabelle	Dunn	annabelledunn@examplecom	5151234444	2017-09-16	2	Administration Assistant
102	Emma	Perkins	emmaperkins@examplecom	5151235555	2017-02-16	1	Marketing Manager
103	Amelie	Hudson	ameliehudson@examplecom	6031236666	2017-09-16	102	Marketing Representative

104	Harper	Spencer	harperspencer@examplecom	5151237777	2007-06-16	2	Human Resources Representative
105	Gracie	Gardner	graciegardner@examplecom	5151238888	2007-06-16	2	Public Relations Representative
106	Rose	Stephens	rosestephens@examplecom	5151238080	2007-06-16	2	Accounting Manager
107	Summer	Payne	summerpayne@examplecom	5151238181	2007-06-16	106	Public Accountant
2001	John	Doe	john.doe@example.com	1234567890	2023-01-01	NULL	Manager
2002	Jane	Smith	jane.smith@example.com	8987654321	2023-02-01	2001	Salesperson
2003	Bob	Johnson	bob.johnson@example.com	5551234	2023-03-01	2001	Warehouse Staff
2004	Alice	Williams	alice.williams@example.com	5555678	2023-04-01	2002	Customer Service

111 rows in set (0.09 sec)

（20）分析步骤（14)、(18）和（19）的执行结果并给出理由

步骤如下：

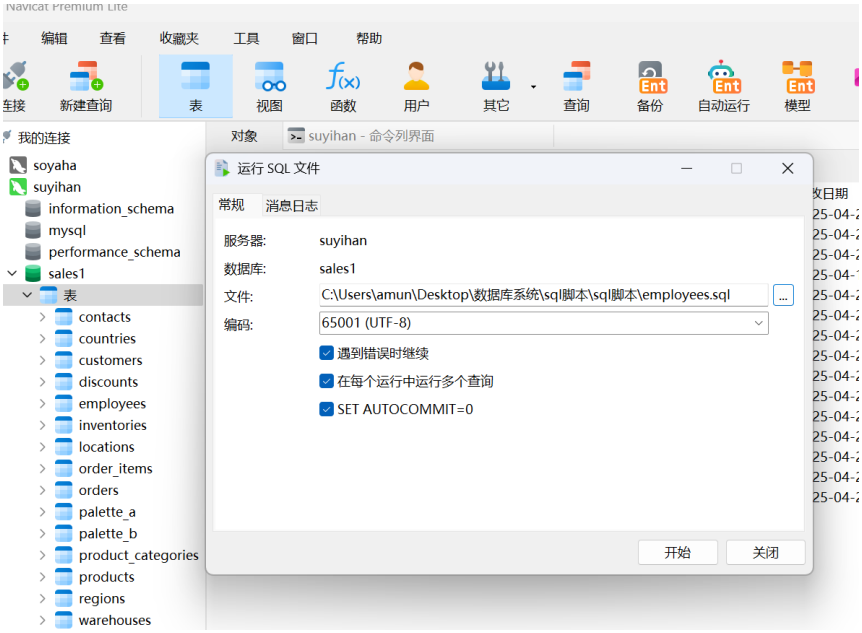
14 步查询 employees 表中 manager_id 为 1 的记录，正常执行，

18 步在执行删除命令后再查询 employees 表中 manager_id 为 1 的记录，应该是想看出删除是否成功，这里失败了。

19 步再查询整个表进一步验证了这个操作。

（21）使用所提供的 employees 表的 SQL 脚本将 employees 表复原到原始状态，包括约束和数据

步骤如下：



3.实验总结

3.1 完成的工作

执行 SHOW CREATE TABLE employees; 查看 employees 表的外键约束信息。

对 employees 表的外键约束进行删除与重建操作，包括添加带 ON DELETE CASCADE 选项的外键约束。

执行 DELETE FROM employees WHERE manager_id = 1; 尝试删除特定记录，并观察结果。

多次执行查询操作（如 SELECT * FROM employees WHERE manager_id = 1; 和 SELECT * FROM employees;）验证数据变化。等

3.2 对实验的认识

通过实验，掌握了外键约束的查看（SHOW CREATE TABLE）、删除（ALTER TABLE ... DROP FOREIGN KEY）和重建（ALTER TABLE ... ADD CONSTRAINT ... FOREIGN KEY）方法。理解了 ON DELETE CASCADE 和 ON DELETE RESTRICT 的差异：前者在删除父记录时级联删除子记录，后者阻止删除有子记录依赖的父记录，以保障数据完整性。这让我认识到外键约束在维护数据库数据一致性和完整性中的关键作用，以及不同约束选项在实际业务场景中的适配性。

思考题：设计实例通过创建父表 departments（含主键 dept_id）和子表 employees（外键 dept_id 设置 ON UPDATE CASCADE），插入关联数据后更新父表主键，可见子表外键值自动同步更新，验证了该选项在父表主键变更时会级联更新子表外键、保持数据一致性的作用；

对比 ON UPDATE RESTRICT 场景，其会阻止父表更新并报错，凸显 ON UPDATE CASCADE 在维护外键关联完整性上的自动级联特性。

3.3 遇到的困难及解决方法

“无”。