**Database Concepts Experiment Guide**

**School of Computer Science**

**October 2020**

**History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Ver** | **Date** | **Author** | **Content** |
| Ver 1.0 | 2016.09 | Ning Li | Create according to the Chinese Course |
| Ver 2.0 | 2017.11 | Xiaonan Zhao | Modify some inappropriate contents |
| Ver 3.0 | 2019.03 | Xiaonan Zhao | Add an optional experiment to Ex 2 |
| Ver 4.0 | 2020.10 | Xiaonan Zhao | Modify for MySQL |

Content

[Content](#_Toc23968)

[1. Experiment Info](#_Toc22792)

[1.1. Experiment List](#_Toc9989)

[1.2. Experimental Environment](#_Toc865)

[1.3. Experiment Grade](#_Toc18058)

[1.4. Experiment Requirement](#_Toc12999)

[2. Experimental Preparation](#_Toc20883)

[2.1. Construction of Experimental Environment](#_Toc31892)

[2.2. Experimental Environment Access](#_Toc3988)

[3. Experiment 1: Create and Manage Database and Table](#_Toc23551)

[3.1. Goal](#_Toc25133)

[3.2. Content](#_Toc23110)

[4. Experiment 2: Manipulate the Data in Table](#_Toc11069)

[4.1. Goal](#_Toc16642)

[4.2. Content](#_Toc2150)

[5. Experiment 3: Data Integrity and Security](#_Toc28683)

[5.1. Goal](#_Toc14299)

[5.2. Content](#_Toc31646)

[6. Experiment 4: View and Index](#_Toc22638)

[6.1. Goal](#_Toc7335)

[6.2. Content](#_Toc1843)

[7. Experiment 5: Trigger, Procedure and Function](#_Toc3160)

[7.1. Goal](#_Toc12096)

[7.2. Content](#_Toc28019)

[8. Experiment 6: ODBC/JDBC（optional）](#_Toc17479)

[8.1. Goal](#_Toc14581)

[8.2. Content](#_Toc10012)

[9. Experiment 7：Transaction](#_Toc4354)

[9.1. Goal](#_Toc11119)

[9.2. Content](#_Toc13375)

[10. Experiment 8：Project](#_Toc12344)

[10.1. Goal](#_Toc23811)

[10.2. Requirement](#_Toc2219)

[10.3. Content](#_Toc29445)

[11. Appendix](#_Toc8627)

# Experiment Info

## Experiment List

**Table 1 Experiment List**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Content | Detailed Info | Type  （ratio） | Hours |
| 1 | Create and manage database and table | Backup/Restore database , Create/Alter/Drop database and table | Verification  (5%) | 2 |
| 2 | Data Manipulation of base table | Query, Insert, Delete, Update  Optional： design query statement for TPC-H | Design and verification(10%) | 2 |
| 3 | Data integrity and security control | Data integrity and security | Design and verification(5%) | 2 |
| 4 | View and Index | Create/Drop view and index  Optional：DB maintain and manage manual | Design and verification(10%) | 2 |
| 5 | Trigger, storage procedure and function | Trigger, storage procedure and function  Optional：Bank DB operations practice | Design and Programming  (10%) | 2 |
| 6 | ODBC/JDBC programming | ODBC database programming  (JDBC is optional) | Programming  (10%) | 2 |
| 7 | Transaction and concurrency control | Transaction commit/rollback；isolation and lock, transaction log | verification(10%) | 2 |
| 8 | Project | Design and implement a database application system | Comprehension  (40%) | 2 |
|  |  |  |  | 16 |

Note：The 8th experiment，that is to say project，students should focus on database design in 2 hours experiment class, and finish other works such as programming and testing with their extracurricular time. And also，students can f accomplish a larger project in group.

## Experimental Environment

In this experiment, the server-side database instance of the cloud database GaussDB (for MySQL) have been purchased and prepared in advance, and a local MySQL server will be installed in the laboratory. The client can connect the DB sever in many ways, such as DAS (web version) client, local GUI (such as MySQL Workbench) or mysql command line client.

Remote server: GaussDB (for MySQL) [based on MySQL8.0]

Local server: MySQL8.0.\*

Local client: browser, MySQL Workbench, Mysql command line client

## Experiment Grade

The score of each experiment is 100, and we use the final weighted average grade as the final grade of this course.

|  |  |  |
| --- | --- | --- |
| **Item** | **requirement** | **percent** |
| attendance | Attend the course normally | 10% |
| peview | Report(no template, paper/electronic edition | 10% |
| experiment | The correctness of experiment processes and results | 60% |
| report | Correctly, clearly,completely | 20% |
| Total | | 100% |

## Experiment Requirement

1. Strictly abide by all disciplines of the laboratory.
2. Preview before each experiment, and write a simple report (paper / electronic edition).
3. Students should submit the electronic experimental report for each experiment. Experiment report file name should use such format: DBn\_XXXXXX(StudentNo)\_Name. For example, Li Hua’s student ID is 20181234, his report for the first lab class can be named as DB1\_20181234\_lihua.
4. For each experiment, the student should submit an electronic edition of experiment report(A template will be provided) and the experiment source code. If the source code is included in the experiment report, it needn’t be submitted separately again, otherwise the source code and experiment report should be submitted as a compressed package. The experiment report should include the following contents: experiment content, experiment design, experiment steps, and running results (it can be the output of the program or the screenshot of the running screen. The screenshot should be as small as possible to limit the submitted file size)
5. In the experiment report, the screenshot of all the important steps should be contained.
6. Students should write down all the problems they have encountered during the experiment process, and describe how did they solve the problems.
7. Save all your experiment materials and documents until the grade is published.

# Experimental Preparation

## Construction of Experimental Environment

The server-side environment of cloud database gaussdb (for MySQL) does not need to be built individually. If you want to know the process of purchasing and building database, please refer to the guide to building cloud database gaussdb (for MySQL) (Chinese edition only).

If you want to install MySQL 8.0 locally, you can download the installation package from the official website of MySQL and install it by yourself. Reference document: MySQL 8.0 installation and common mistakes.doc

<https://dev.mysql.com/downloads/windows/installer/8.0.html>

## Experimental Environment Access

1. Web accessing cloud DB GaussDB(for MySQL)

[https://auth.huaweicloud.com/authui/login.html?locale=zh-cn&service=https%3A%2F%2Fwww.huaweicloud.com%2F#/login](https://auth.huaweicloud.com/authui/login.html?locale=zh-cn&service=https://www.huaweicloud.com/" \l "/login)

1. Lohin web page，choose【IAM用户登录】mode，as shown in the following figure。



账号名：hw22222337

IAM用户名(username)： s+studentID，for example：s2018123456（a student）

t+instructorID/studentID,for example: t2020123456（teacher/teach assistant）

IAM用户密码(passwd)：GaussDB@123，（after login, password can be changed）

1. After login normally，the webpage will be shown like following。click【控制台】to enter the console page.



1. In the page of console,choose tha tab of [云数据库GaussDB].



1. Click:”云数据库GaussDB” in the browser，the running DB instance with the name “gauss-nwpu2020” will appear in the window as following figure shown.



1. Use MySQL Workbench or command line to access the remote DB instance。

Hostname: 139.9.119.34

（the purchased internet IP，if it is changed,we will notify in advance）

Username: same to the IAM username (s+studnetID)

Password: GaussDB@123

# Experiment 1: Create and Manage Database and Table

## Goal

1. Familiar with command line and GUI connection method in MySQL.
2. Master SQL statement to create database and table.
3. Master the update and deletion methods of database and tables.
4. Master the basic methods of backup and restore database.
5. Understand the logical structure and physical structure of MySQL database.

## Content

1. Use GUI to connect the DBMS（10 points）。

Hint: try to use both web and MySQL Workbench methods.

1. Use command line to connect the DBMS（10 points）.
2. Create,backup,drop and restore database and tables through GUI.(40 points, each operation 5 points)

* **Database and table**

Database name：SPJ\_MNG，four tables in the database：S, P, J, SPJ

S (SNO, SNAME, STATUS, CITY)

P (PNO, PNAME, COLOR, WEIGHT)

J (JNO, JNAME, CITY)

SPJ (SNO, PNO, JNO, QTY)

The supplier table S is composed of supplier code (SNO), supplier name (SNAME), supplier status (STATUS) and supplier city (CITY).

Part list P consists of part code (PNO), part name (PNAME), color (COLOR) and weight (WEIGHT).

Project table J consists of project code (JNO), project name (JNAME) and project city (CITY).

The supply situation table SPJ is composed of supplier code (SNO), part code (PNO), project code (JNO) and supply quantity (QTY). It indicates that the quantity of a certain part supplied by a supplier to an project is QTY.

Now there are several data in the database as follows. The specified operation is completed based on the database table.

Table S Table SPJ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SNO | SNAME | STATUS | CITY |  |  | SNO | PNO | JNO | QTY |
| S1 | JINGYI | 20 | Tianjin |  | 1 | S1 | P1 | J1 | 200 |
| S2 | SHENGXI | 10 | Beijing |  | 2 | S1 | P1 | J3 | 100 |
| S3 | DONGFANGHONG | 30 | Beijing |  | 3 | S1 | P1 | J4 | 700 |
| S4 | FENGTAISHENG | 20 | Tianjin |  | 4 | S1 | P2 | J2 | 100 |
| S5 | WEIMIN | 30 | Shanghai |  | 5 | S2 | P3 | J1 | 400 |
| Table P |  |  |  |  | 6 | S2 | P3 | J2 | 200 |
| PNO | PNAME | COLOR | WEIGHT |  | 7 | S2 | P3 | J4 | 500 |
| P1 | nut | red | 12 |  | 8 | S2 | P3 | J5 | 400 |
| P2 | bolt | green | 17 |  | 9 | S2 | P5 | J1 | 400 |
| P3 | screwdriver | blue | 14 |  | 10 | S2 | P5 | J2 | 100 |
| P4 | screwdriver | red | 14 |  | 11 | S3 | P1 | J1 | 200 |
| P5 | cam | blue | 40 |  | 12 | S3 | P3 | J1 | 200 |
| P6 | gear | red | 30 |  | 13 | S4 | P5 | J1 | 200 |
| Table J |  |  |  |  | 14 | S4 | P6 | J3 | 100 |
| JNO | JNAME | CITY |  |  | 15 | S4 | P6 | J4 | 300 |
| J1 | SANJIAN | Beijing |  |  | 16 | S5 | P2 | J4 | 100 |
| J2 | YIQI | Changchun |  |  | 17 | S5 | P3 | J1 | 200 |
| J3 | Spring Factory | Tianjin |  |  | 18 | S5 | P6 | J2 | 200 |
| J4 | Shipyard | Tianjin |  |  | 19 | S5 | P6 | J4 | 500 |
| J5 | Locomotives Factory | Xian |  |  |  |  |  |  |  |
| J6 | Radio Factory | Changzhou |  |  |  |  |  |  |  |
| J7 | Semiconductor Factory | Nanjing |  |  |  |  |  |  |  |

* + 1. Create database SPJ\_MNG

Hint：right click【Create Schema】

* + 1. Create four tables in database SPJ\_MNG（add some tuples in each table)。

Hint：choose one created talble, right click and choose[select rows]，in the display results, there is a editable table, input some tuples directly.

* + 1. Export the database SPJ\_MNG as a \*.SQLfile

Hint：in MySQL WorkBench upper left a navigation can be used，choose Administration → Data Export。Choose the target database, set some parameters and export it.

* + 1. Delete the table of supplier(table SPJ)

Hint：【Drop Table】

* + 1. Delete database SPJ\_MNG

Hint：【Drop Schema】

* + 1. Restore the database SPJ\_MNG with the file you have backed up in step （3）

Hint：in MySQL WorkBench choose Administration → Data Import/Restore。Choose the database SPJ\_MNG first，then choose all the the directory to import.

* + 1. Update table S，Add an attribute of contact phone number STEL, the data type is string , and modify the maximum string length allowed by SNO in table S.
    2. Understand the physical storage files of MySQL, and check the data files under the local MySQL service installation directory (such as the default installation directory: C: \programdata \ MySQL\ MySQL server 8.0\ data). Try to create tables according to different storage engines of InnoDB and MyISAM, observe and explain the differences of physical storage files.

Hint: you can query the data storage directory in MySQL:

show global variables like "%datadir%";

1. Create,backup,drop and restore database and table using MySQL command line.(40 points, 5 points for each requirement)

* Database and tables

Datbase name: university

Tables:

(primary key: red color foreign key: )

Classroom:building, room\_number,capacity

Department:dept\_name, building, budget

Course:course\_id,title,dept\_name,credits

Instructor:ID,name,dept\_name,salary

Section:course\_id,sec\_id,semester,year,building,room\_number,time\_slot\_id

Teaches:ID,course\_id,sec\_id,semester,year

Student:ID,name,dept\_name,tot\_cred

Takes:ID,course\_id,sec\_id,semester,year,grade

Advisor:s\_ID,i\_ID (s\_ID references student (ID), i\_ID references instructor (ID))

Time\_slot: time\_slot\_id,day,start\_hr,start\_min,end\_hr,end\_min

Prereq: course\_id, prereq\_id

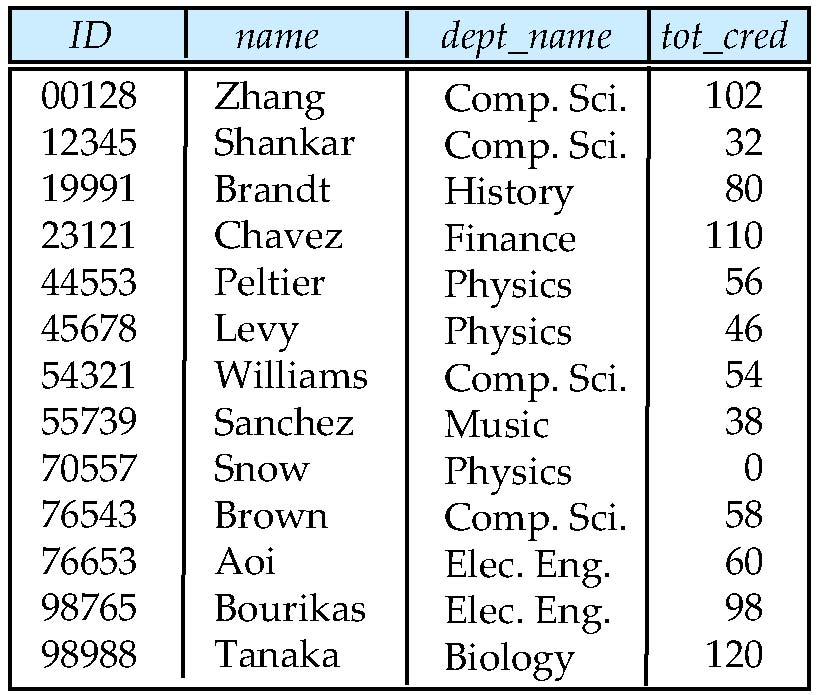
Note: some hint for data type and domain

*building:varchar(15), room\_number: varchar(7), capacity:numeric(4,0)*

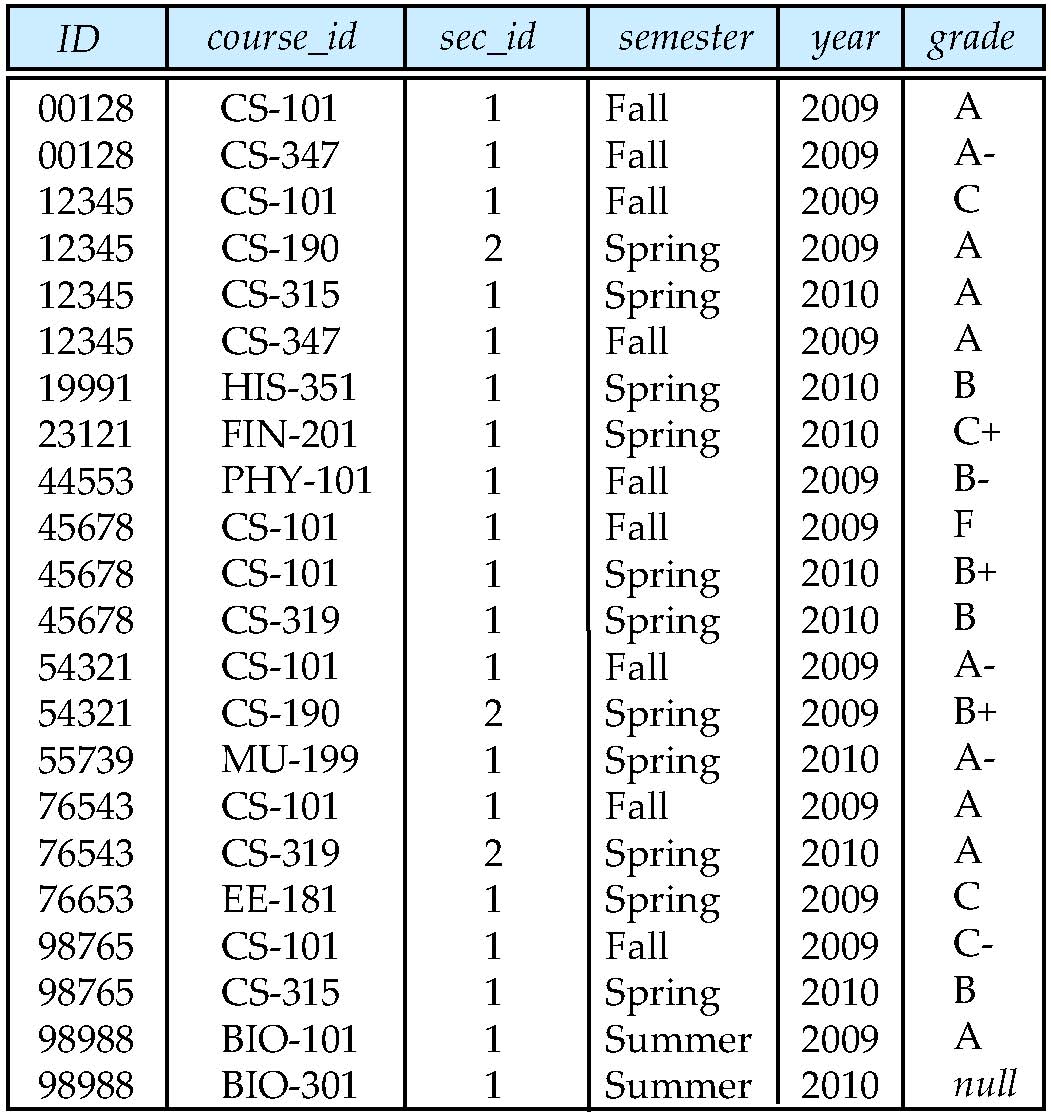
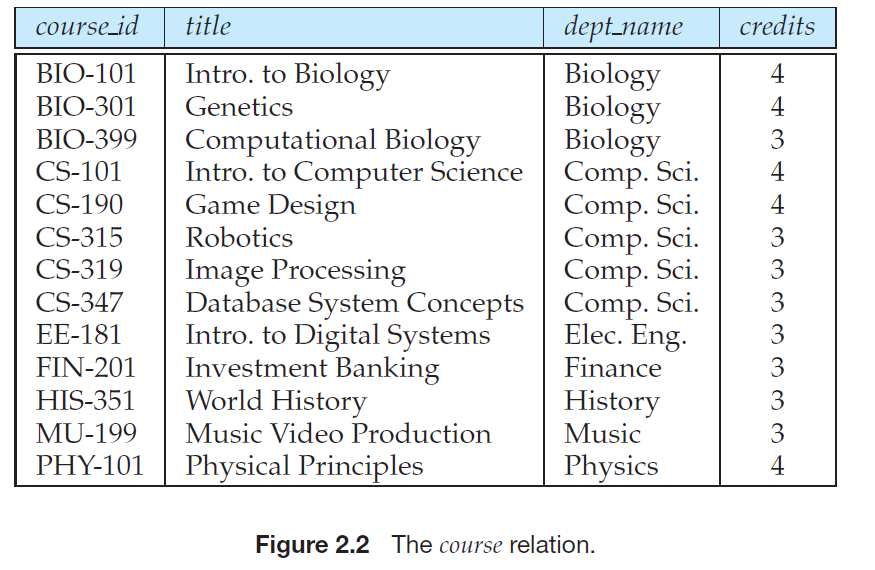
*name/dept\_name:varchar(20), budget: numeric(12,2), course\_id:varchar(8),*

*title:varchar(50), credits:numeric(2,0), ID:varchar(5), salary:numeric(8,2),*

*sec\_id:varchar(8), semester:varchar(6),year:numeric(4,0),time\_slot\_id:varchar(4), tot\_cred:numeric(3,0),grade: varchar(2), day:varchar(1),start\_hr:numeric(2)*



**The *student* relation**



**The *takes* relation**

1. Use SQL statements to create a database of university.
2. Use SQL to create 3 tables:student, course, takes, define the data type and primary key, ignore the other constraints, add some tuples if you like.
3. Backup the database of university.

Hint: in the command line window, swith to the directory C:\Program Files\MySQL\MySQL Server 8.0\bin(if the directory is already set in the environment variable, skip it), use the command of mysqldump.

mysqldump -h localhost -u root -p university> d:\university.sql

mysqldump -h localhost -u root -p --no-data --databases university> d:\s2.sql

-h: server name or IP；-u: username; -p：password university：database to be back up

--no-data: database not to be back up

Meaning of the command： backup the local database ‘university’ to the file ‘d:\university.sql’ 。

For more information of this command, refer to the url：<https://dev.mysql.com/doc/refman/8.0/en/mysqldump.html>

With command mysqldump, we can also export the database as txt or csv files.

1. Delete created tables with SQL statement.
2. Delete created database with SQL statement.
3. Restore the database with the backed up files you’ve got in the operation of step (3).

Hint: first, use SQL statement to create university database, and then use the backup files in (3) to restore. open cmd.exe and run the program, switch to the bin directory of MySQL (if the environment variables have been configured, no switch), and use the following command to restore:

mysql -h localhost -u root -p university< d:\ university.sql

If you have a prepared data CSV file, you can also use the command of LOAD DATA to import the data.

1. Use SQL statement to add a new column STEL to store phone number into the table of student, and modify the max length of the attribute ID.
2. Check the SQL scripts that define the database or table in the SQL file generated by mysqldump, and compare the similarities and differences between the automatically generated scripts and the SQL statements written by yourself.

# Experiment 2: Manipulate the Data in Table

## Goal

1. Master all kinds of data operation about basic table in GUI.

2. Familiar with SQL statements for data insertion, modification and deletion of basic tables.

3. Master the SQL statement of data query.

4. Master the basic knowledge of SQL query performance analysis.

5. Understand TPC-H benchmark database.

## Content

1. Use SQL statement to insert all the tuples into the database SPJ\_MNG and university which have been list in the previous experiment.
2. Modificaion the data of tables with SQL statement.
3. Modify one tuple in the table of student
4. Delete one tuple from table of student.
5. In the database of SPJ\_MNG, use SQL statement to do the following update operations:
6. Change the color of all red parts to blue.
7. Part P6 supplied by S5 for J4 is replaced by S3, please make necessary modification.
8. Delete S2 record from supplier table and delete corresponding record from supply table.
9. Please insert (S2, J6, p4200) into the supply table SPJ.Finish the following queries about the database university with SQL statement.
10. Use three different ways (SQL statement) to find the student ID and name of all students who take “Database System Concept”, and then analyze and compare the performance of each query process.
11. For university database, complete the following data query with SQL statement
12. Query the total score of credits obtained by each student , and output the student ID, name and credit obtained in the order from high to low.
13. Query the name of the student: the student has taken all courses and one of the courses has a grade of better than B .
14. Use at least three different SQL statements to query the university database: query the student ID and name of the course named "database", and then design the experiment by ourselves, compare and analyze the efficiency of the three kinds of query with data, and analyze the reasons.

Hint: in order to compare it more clearly, you’d better create a bigger table.

1. （optional）TPC-H is one of the database benchmarks released by TPC international organization. The database simulates the data of a typical enterprise: parts, customers, parts, suppliers, products, orders and so on- ddl.sql Documents.Based on database TPC-H (database definition statements can be defined referring to the file “tcp-h-ddl.sql”), design the following queries and test the queries with some data：
2. Single table query (to realize operations of projection and selection)
3. Grouping stastistic query(using “group by”, without using “group by”)
4. Single table query with self-join operation
5. Multple table query with join operation
6. Nested query with IN clause
7. Nested query with EXISTS clause
8. Neseted query with FROM　clause
9. Set query( intersect, union and except)

**The TPC-H database is designed as following:（referring to the file tcp-ds-v2.17.3.docx）：**



Discription：

1. The database models a company which includes the information of :customers, products, supplier, orders, partsupp, etc.
2. The numbes or SF(Scale factor)\* numbers under each table are used to indicate the expected count of tuples of the table, the detailed information of tables are as following:

|  |  |  |
| --- | --- | --- |
| **PART Table** |  |  |
| Column Name | Datatype Requirements | Comment |
| P\_PARTKEY | identifier | SF\*200,000 are populated |
| P\_NAME | variable text, size 55 |  |
| P\_MFGR | fixed text, size 25 |  |
| P\_BRAND | fixed text, size 10 |  |
| P\_TYPE | variable text, size 25 |  |
| P\_SIZE | integer |  |
| P\_CONTAINER | fixed text, size 10 |  |
| P\_RETAILPRICE | decimal |  |
| P\_COMMENT | variable text, size 23 |  |
| Primary Key**:** P\_PARTKEY | | |
|  |  |  |
| **SUPPLIER Table** | |  |
| Column Name | Datatype Requirements | Comment |
| S\_SUPPKEY | identifier | SF\*10,000 are populated |
| S\_NAME | fixed text, size 25 |  |
| S\_ADDRESS | variable text, size 40 |  |
| S\_NATIONKEY | Identifier | Foreign Key to N\_NATIONKEY |
| S\_PHONE | fixed text, size 15 |  |
| S\_ACCTBAL | decimal |  |
| S\_COMMENT | variable text, size 101 |  |
| Primary Key**:** S\_SUPPKEY | | |
|  |  |  |
| **PARTSUPP Table** | |  |
| Column Name | Datatype Requirements | Comment |
| PS\_PARTKEY | Identifier | Foreign Key to P\_PARTKEY |
| PS\_SUPPKEY | Identifier | Foreign Key to S\_SUPPKEY |
| PS\_AVAILQTY | integer |  |
| PS\_SUPPLYCOST | Decimal |  |
| PS\_COMMENT | variable text, size 199 |  |
| Primary Key**:** PS\_PARTKEY, PS\_SUPPKEY | | |
| **CUSTOMER Table** | | |
| Column Name | Datatype Requirements | Comment |
| C\_CUSTKEY | Identifier | SF\*150,000 are populated |
| C\_NAME | variable text, size 25 |  |
| C\_ADDRESS | variable text, size 40 |  |
| C\_NATIONKEY | Identifier | Foreign Key to N\_NATIONKEY |
| C\_PHONE | fixed text, size 15 |  |
| C\_ACCTBAL | Decimal |  |
| C\_MKTSEGMENT | fixed text, size 10 |  |
| C\_COMMENT | variable text, size 117 |  |
| Primary Key**:** C\_CUSTKEY | |  |
|  |  |  |
| **ORDERS Table** |  |  |
| Column Name | Datatype Requirements | Comment |
| O\_ORDERKEY | Identifier | SF\*1,500,000 are sparsely populated |
| O\_CUSTKEY | Identifier | Foreign Key to C\_CUSTKEY |
| O\_ORDERSTATUS | fixed text, size 1 |  |
| O\_TOTALPRICE | Decimal |  |
| O\_ORDERDATE | Date |  |
| O\_ORDERPRIORITY | fixed text, size 15 |  |
| O\_CLERK | fixed text, size 15 |  |
| O\_SHIPPRIORITY | Integer |  |
| O\_COMMENT | variable text, size 79 |  |
| Primary Key**:** O\_ORDERKEY | | |
|  |  |  |
|  | | |
| **LINEITEM Table** | |  |
| Column Name | Datatype Requirements | Comment |
| L\_ORDERKEY | identifier | Foreign Key to O\_ORDERKEY |
| L\_PARTKEY | identifier | Foreign key to P\_PARTKEY, first part of the compound Foreign Key to (PS\_PARTKEY, PS\_SUPPKEY) with L\_SUPPKEY |
| L\_SUPPKEY | Identifier | Foreign key to S\_SUPPKEY, second part of the compound Foreign Key to (PS\_PARTKEY, PS\_SUPPKEY) with L\_PARTKEY |
| L\_LINENUMBER | integer |  |
| L\_QUANTITY | decimal |  |
| L\_EXTENDEDPRICE | decimal |  |
| L\_DISCOUNT | decimal |  |
| L\_TAX | decimal |  |
| L\_RETURNFLAG | fixed text, size 1 |  |
| L\_LINESTATUS | fixed text, size 1 |  |
| L\_SHIPDATE | date |  |
| L\_COMMITDATE | date |  |
| L\_RECEIPTDATE | date |  |
| L\_SHIPINSTRUCT | fixed text, size 25 |  |
| L\_SHIPMODE | fixed text, size 10 |  |
| L\_COMMENT | variable text size 44 |  |
| Primary Key**:** L\_ORDERKEY, L\_LINENUMBER | | |
|  |  |  |
| **NATION Table** |  |  |
| Column Name | Datatype Requirements | Comment |
| N\_NATIONKEY | identifier | 25 nations are populated |
| N\_NAME | fixed text, size 25 |  |
| N\_REGIONKEY | identifier | Foreign Key to R\_REGIONKEY |
| N\_COMMENT | variable text, size 152 |  |
| Primary Key**:** N\_NATIONKEY | | |
|  |  |  |
| **REGION Table** |  |  |
| Column Name | Datatype Requirements | Comment |
| R\_REGIONKEY | identifier | 5 regions are populated |
| R\_NAME | fixed text, size 25 |  |
| R\_COMMENT | variable text, size 152 |  |
| Primary Key**:** R\_REGIONKEY | |  |

# Experiment 3: Data Integrity and Security

## Goal

1. To practice how to define the data integrity.
2. To practice how to create users
3. To practice how to grant/revoke privileges of databases and tables.

## Content

Database name：Student， including 3 tables：

S (SNO, SNAME, SGENDER, SBIRTH, SDEPT, SAGE)

C (CNO, CNAME, CPNO, CREDIT)

SC (SNO, CNO, GRADE)

Student information table S consists of student number (SNO), name (SNAME), gender (SGENDER), date of birth (SBIRTH), deparatment(SDEPT), age (SAGE).

Course information table C consists of course number (CNO), course name (CNAME), prerequisite course number (CPNO) and credit.

SC is composed of SNO, CNO and GRADE, it indicates that a student has taken a course with grade.

Table S Table SC

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SNO | SNAME | SGENDER | SBIRTH | SDEPT |  | SNO | CNO | GRADE |
| 2001 | Yong Li | male | 2000/01/01 | MA |  | 2001 | 1 | 92 |
| 2002 | Liu Chen | female | 2001/02/01 | IS |  | 2001 | 2 | 85 |
| 2003 | Wang Min | female | 1999/10/01 | CS |  | 2001 | 3 | 90 |
| 2004 | Zhang Li | male | 2001/06/01 | IS |  | 2002 | 2 | 78 |
|  | Table C |  |  |  |  | 2002 | 3 | 84 |
| CNO | CNAME | CPNO | CREDIT |  |  | 2003 | 6 | 91 |
| 1 | Database | 2 | 3 |  |  |  |  |  |
| 2 | Math |  | 5 |  |  |  |  |  |
| 3 | Information system | 1 | 2 |  |  |  |  |  |
| 4 | Operating system | 5 | 3 |  |  |  |  |  |
| 5 | Data structure | 6 | 3 |  |  |  |  |  |
| 6 | C languange |  | 2 |  |  |  |  |  |

1. Add the following constraint and index with GUI(for university database)（10 points）.
   * 1. Not null：add the not null constraint to **S**(SBITH).
     2. Primary key：set the SNO as the primary key.
     3. Unique constraint：add unique constraint for the primary key of SNAME, the constraint name is uk\_Sname.
     4. Default constraint：add the default value to **S**(SGENDER), the default value is “male”.
2. Add forgein key constraints to table **SC** in the database **Student** with GUI, set SNO(foreign ke name is: sc\_fk\_sid) as a foreign key referencing table S, and set CNO as another foreign key referencing table C ***,*** name it with sc\_fk\_ cno. Try and validate different strategies in violation of the foreign key constraints :

NO Action/restrict/cascade/set to null（10 points）

1. Drop the three tables in database **Student**, and create some tables through SQL statements with the following constraints.(10 points）

* Table ***S***：same to the constraints set in question 1(1)。
* Table ***C***：set CPNO as a foreign key，referencing table c itself with the attribute CNO.
* Table **SC**：set the foreign key constraints same to question2, and set the valid range of attribute GRADE with[0,100]. In addition, add one attribute ID to table SC, and set it as a primary key, and it can increase automaticlly. When a new tuple is inserted to the table, its(ID) value will increase by +1.

1. Add or remove the following integrity constraints with SQL language. (10 points)

(1) Add constraint: SGENDER ’s value in table S can only be "male" or "female".

(2) Delete the SGENDER value constraint created in table S.

(3) Remove the foreign key constraints from the SC table.

(4) Add a new column TNAME (indicating the name of the student's tutor) in the student table S, and require that the tutor name must be all uppercase or lowercase letters, and the length cannot be less than 8 characters (Hint: You can use the Length() function and regular expression to CHECK your character string).

Try to finish the following requirements by connecting to the local database server

1. Create and authorize new users in GUI. (20 points)

(1) Create two users who can access the current student database: Wang Ming and Li Yong.

(2) Complete the following authorizing:

① User Wang Ming has the priviliges to select and insert all tables.

② User Li Yong has the priviliges of select, insert, delete, update and create on the database.

2. Use SQL statement to authorize and withdraw permissions and verify permissions. (40 points)

Requirements: create relevant users and specified database tables, complete authorization and authority verification for each question, and then withdraw the authority and verify the authority.

(1) User Wang Ming has query privilege on two tables.

(2) User Li Yong has insert and delete privileges on the two tables.

(3) Each instructor only has the right to query his own record.

(4) User Liu Xing has query privilege on employee table and update privilige on salary field.

(5) User Zhang Xin has the right to modify the structure of the two tables.

(6) User Zhou Ping has all permissions on the two tables, and has the authority to authorize other users.

(7) User Yang Lan has the privilige to query the maximum wage, minimum wage and average wage from each department employee, but he cannot view the salary of each person.

Hint:

1. Create user SQL statment: create user 'Wang Ming' @'localhost 'identified by' 123456 ';( in command line, ' is the quotation mark)

User name: Wang Ming @ localhost; password: 123456

1. You can use the **show grants for username** （including the location: for example @localhost）to view the privileges list of each user.
2. Create an appropriate view for some questions in this part.
3. To view system current user: system function user ()

# Experiment 4: View and Index

## Goal

1. Be familiar with the use of GUI and SQL language to create, update and delete views.

2. Proficient in creating and deleting index using GUI and SQL language.

3. Understand and verify the role of index.

## Content

1. In the student database, use SQL statement to create a view of students who have taken the database course and are born in 2001. The view includes the information of student number, name, gender and grade. (5 points)
2. Create a view for the supply situation of “SANJIAN” project, including the attributes of supplier code (SNO), part code (PNO) and supply quantity (QTY), with two ways with SQL statement.(view name: V\_SPJ) (10 points)
3. Complete the following view query with SQL statement. (10 points)
4. Find out all the parts code and their quantity used by “SANJIAN” project.
5. Find out the supply situation of supplier S1.
6. Update the data of views with SQL statement.(15 points)
   * 1. Insert a tuple into the view V\_SPJ.

Hint:

1. In table SPJ, when JNO is permitted to be null, the tuple can be inserted into the actual table. Because of JNO is NULL，the tuple will not appears in the view.
2. When the attribute JNO in table SPJ was set to be not null, you can use ‘instead of’ trigger. MySQL does not support to create a trigger on a view, so in the MySQL environment, it is not necessary to answer this question.
   * 1. Modify the quantity value of any tuple in the view V\_SPJ.
     2. Delete one tuple from the view V\_SPJ

Hint: Only when the created view can be resoluted, it can be deleted normally, otherwise it will fail to be deleted.

1. Create a descending index named IX\_CNo for the CNO attribute of C table in student database by using GUI. (5 points)
2. Use SQL statement to complete the following index operation on student database. (15 points)

(1) Create a non-unique index named IX\_CNAME on the CNAME attribute of table C.

(2) Create a composite index named IX\_ngd\_NGD on the table S, which is an ascending index for sname, sgender and sdept attribute sets.

Hint: you can use show index from table name to view the index information on the table.

(3) Delete index IX\_ CNo of table C。

(4) Based on the above indexes (table C: primary key index of CNO, general index of CNAME; table S: primary key index of SNO, IX\_ NGA composite index), use explain statement to obtain the query plan of each query statement, to observe the index usage in each query statement.

* + 1. explain select \* from c;
    2. explain select \* from c where cno = ‘1’;
    3. explain select \* from c where cname=’database’ ;
    4. explain select \* from c where cname like ‘%database%’;
    5. explain select \* from c where cname like ‘database%’;
    6. explain select \* from s where sname ='Zhangli' and sno='2001';
    7. explain select \* from s where sname ='Zhangli' and sgender='male' and sdept='IS';
    8. explain select \* from s where sname ='Zhangli' and sgender='male';
    9. explain select \* from s where sname ='Zhangli';
    10. explain select \* from s where sgender ='male';
    11. explain select \* from s where sgender ='male' and sdept='IS';

1. Suppose there is a basic table userinfo as follows, design an experiment to verify the effect of index on database query efficiency. (40 points) create table userinfo

(

user\_id int primary key, //USER ID

username varchar(10), //USERNAME

gender char(1), //GENDER

age int, //AGE

c\_id int //NO OF COLLEGE

)

(1) Verify the efficiency difference between indexed and non-indexed queries.

(2) Verify the query efficiency of single field narrow index and multi field wide index, pay attention to understand the left most matching principle in wide index.

(3) Verify the difference of query efficiency between clustered index (primary key index) and secondary index: build clustered index and nonclustered index on the same field to compare query efficiency. (optional)

(4) At present, only memory engine of MySQL supports both b-tree index and hash index. Create a basic table based on memory storage engine in mysql, and verify the query efficiency difference between b-tree index and hash index based on this table. (optional)

Tips and requirements:

(1) Import a large amount of data (at least 100millio tuples) for validation.

(2) Use the **explain** command to analyze whether the SQL query uses the created index, and then analyze the performance reasons based on it.

(3) Generally, b-tree index is suitable for range query and hash index is suitable for point query.

Optional experiments

4. Refer to the "gaussdb (for MySQL) database maintenance and management manual.docx", and understand the following functions on the DAS side of Huawei cloud database:

(1) Section 1.2 manages the database cluster.

Special attention: this section of the experiment only to view the operation! Adding cluster nodes, changing cluster specifications, resetting administrator account name and password, cluster backup and recovery, parameter modification and other changes that have an impact on the global situation are only try to understand, not do the actual operations.

(2) 1.3.1 performance monitoring.

(3) 1.3.2 performance tuning.

# Experiment 5: Trigger, Procedure and Function

## Goal

To practice how to create the trigger/procedure/function.

## Content

1. For SPJ\_ MNG database, create and execute the following procedures. (40 points)
2. Create a stored procedure with no parameters named jsearch1. The function of the procedure is: when the procedure is executed, it returns all the information of Beijing suppliers in table S. Call the procedure and verify the results. (5 points)
3. Create a procedure with input parameters named jsearch2. The function of the procedure is: when you enter the city name of a supplier (such as Beijing), it will return all the information of the specific supplier. Call the procedure and verify the results. (5 points)
4. Create a procedure (or a function) jsearch3 with input and output parameters. The function of this procedure/function is: when a supplier number (input parameter SNO) is entered, the name of the supplier (output parameter SNAME) will be returned. Call the procedure/function and verify the results. (5 points)
5. Create a procedure jsearch4 using cursors, call the procedure and verify the result after. The function of the procedure is: when a project number JNO is input, the names (SNAMEs) of all suppliers supplying the engineering parts will be returned. These supplier names are spliced into a string and separated by comma ','.

For example: input: J2, output: ‘JINGYI, SHENGXIN, WEIMIN'. (10 points)

1. View the text information of the procedures jsearch1 and jsearch2. (5 points)

Hint: use the command ‘show create procedure jsearch1’. For convenience of viewing, you can add \G at the end of the above command and display it in the form of two columns of text.

1. View the basic status information of the procedures jsearch1 and jsearch2. (5 points)

Hint: Show procedure status like 'jsearch%';

1. Delete the procedure jsearch1. (5 points)
2. Create and execute the following triggers for student database: (40 points in total)
3. Delete the foreign key constraint on the SC table and create INSERT trigger named insert\_s for the table SC. The function of the trigger is: when the user inserts a record into the SC table, if the inserted CNO value is not the existing value of CNO in table C, the user will be prompted a message "the data not in table C cannot be inserted" and the insertion of the data will be prevented; if the inserted SNO value is not an existing value of SNO in table S, the user will be prompted that "the data not in table S cannot be inserted" and the insertion of the data will be prevented. After the trigger created successfully, insert some records into the SC table to verify whether the trigger works normally. (5 points)
4. Create a DELETE trigger named dele \_ s1 for table S. The function of the trigger is: to prompt the user a message of "the data in this table cannot be deleted" and prevent the user from deleting the data in table S. After the trigger is created successfully, delete some record in table S to verify whether the trigger works normally. (5 points)
5. Create a DELETE trigger named dele \_ s2 for the table S, the function of the trigger is to delete the his course selection record in SC table when deleting a student’s record in S table. After the trigger is created successfully, delete some records in the table S and verify whether the trigger works normally (confirm whether the relevant data of S table and SC table are both deleted). (5 points)
6. Create an UPDATE trigger for table S named update\_s, the function of this trigger is to prohibit updating the contents of the column "sdept" in table S (make the updating be failed and prompt a message of "the column of ‘sdept’ cannot be updated). After the trigger is created successfully, update the content of the "sdept" column in the table S to verify whether the trigger works normally. (5 points)
7. Delete update\_s trigger. (5 points)
8. Design a before update trigger and after update trigger, and compare the difference between the two triggers. (5 points)
9. Create a new curriculum score statistics table CAVGGRADE (CNO, snum, examsnum, avggrade), for the columns are indicating the course number, the number of students who choose the course, the number of students taking the exam, and the average score of the course respectively. Use a trigger to achieve the following functions: when insert, delete or update a person's score of SC table, automatically update the records in table CAVGGRADE correspondingly. Note that if the grade of a student in SC table is NULL, it indicates that the student has not taken the exam, and it is not necessary to calculate the average score. However, when grade is 0, that is, the score of the student is 0, the average grade needs to be calculated. (10 points)

Hint: in mysql, you need to create three triggers for insert, update and delete actions. You can first design and implement a common procedure, and then call the stored procedure in the 3 triggers.

1. Create an table employee (EID, ename, salary) for the employees, assuming there are 1000 employee data in the table, complete the following requirements. (total 20 points, 10 points for each question)
2. In order to automatically generate 1000 employee data, a user-defined function GENERATEEID is created to automatically generate employee ID. The employee ID is required to be an 8-digit number. The higher four digits indicate the current year in which the employee data is inserted, and the last four digits increase in the order from 0001 to 9999. For example, the first record inserted in the year of 2015 is 20150001, and all 1000 employee IDs are 20150001-20151000. Call this function to insert 1000 pieces of data automatically. (Note:when inserting data that the employee name can be any value , and the value of salary is between 2000 and 5000)
3. The company plans to increase the salary of each employee according to certain rules. Please use cursor to create a procedure and execute the procedure to complete the salary adjustment. The salary increase rules are as follows:

* The salary below 3000 yuan, with a monthly increase of 300 yuan;
* The salary between 3000-4000 yuan, with a monthly increase of 200 yuan;
* The salary more than or equal to 4000 yuan, with a monthly increase of 50 yuan;

# Experiment 6: ODBC/JDBC（optional）

## Goal

To practice how to access the database from applications with ODBC/JDBC or other methods.

## Content

1. ODBC data source configuration and program debugging. (20 points)
2. To configure an ODBC data source, the data source name is: student, which contains the S table (student information table).

Hint: when configuring ODBC data source, you need to pay special attention to whether it is 32-bit or 64 bit. The configuration information should be consistent with the 32 / 64 bit of the application.

1. To understand ODBC programming, read and run the program example (MFC or CSharp code) given in the experiment, it is required to write your own understanding of the program, or draw a flow chart of the program, and give the screenshot of the program running results.

Hint: the example program is implemented by MFC or CSharp language, and can run normally under vs2019. You can also use your favorite programming language or other IDE to re implement, but the function must be the same as the given example program.

1. Refer to the above ODBC example program, use ODBC programming technology, write a simple program, including the database SPJ\_ MNG connection, query, insert, modify and delete. (30 points)
2. Comprehensive application experiment of bank finance. (50 points)
3. According to the requirements, write SQL statements of different scenarios for the bank database.
4. Referring to the programming examples of JDBC / ODBC / third-party library, use one of the ways to access the bank database from the application program, and execute the SQL statement of question (1).

The scenario description of specific banks is as follows:

Suppose that there are five basic entities for the business of Bank C: customers, bank cards, financial products, insurance and funds. For these entities, it is assumed that bank C has the following businesses:

A customer can apply for multiple bank cards.

A customer can purchase multiple financial products, and each type of financial product can be purchased by multiple customers.

A customer can purchase multiple funds, and the same type of fund can be purchased by multiple customers.

A customer can purchase multiple insurance, and the same type of insurance can be purchased by multiple customers

According to the business relationship of Bank C, the following ER diagram is obtained.



图 8-1 ER diagram of C Bbank System Finance

* A complete description of the five entities in Bank C is as follows:

Customer: customer number, customer name, customer ID, customer phone number, customer login password

Bank card: bank card number, bank card type, customer number, balance

Financial products: product name, product number, product status, product price, time period, close start date, sale start date, product discription

Insurance: Insurance name, insurance number, insurance price, applicable population, insurance period, product status

Fund: fund name, fund number, fund type, fund price, risk level, fund manager, fund status

* The tables of all entities in the database finance are as follows:

Customer table: customer (C\_ ID，C\_ NAME，C\_ ID\_ CARD，C\_ PHONE，C\_ PASSWORD）

Card table: Bank\_ card（B\_ NUMBER，B\_ TYPE，B\_ C\_ ID, B\_ BALANCE）

Financial products table:finances\_ product（P\_ NAME，P\_ ID，P\_ DESCRIPTION，P\_ PRICE，P\_CLOSE\_DATE, P\_SALE\_DATE, P\_STATUS, P\_ YEAR）

Insurance table:insurance( I\_ NAME，I\_ ID，I\_ PRICE，I\_ PERSON，I\_ YEAR，I\_ STATUS）

Fund table: fund (F\_ NAME，F\_ ID，F\_ TYPE，F\_ PRICE，RISK\_ LEVEL，F\_ MANAGER, F\_STATUS）

* The purchase relationships between different entities in Bank C are described as follows:

Financial products purchase table: customer number, financial product number, purchase time,purchase quantity, cumulative income, payment bank card number

Insurance purchase table: customer number, insurance number, insurance time, purchase quantity, cumulative income, payment bank card number

Fund purchase table: customer number, fund number, time of fund purchase, purchase quantity, cumulative income, payment bank card number

* The corresponding table of purchase relationships in database finance are as follows:

Financial products purchase table: C\_finances(c\_ ID, P\_ ID, P\_ TIME, P\_ AMOUNT, P\_ INCOME, B\_ C\_ ID)

Insurance purchase table: C\_ insurance (C\_ ID, I\_ ID, I\_ TIME, I\_ AMOUNT, I\_ INCOME, B\_ C\_ ID )

Fund purchase table: C\_ fund (C\_ ID, F\_ ID, F\_ TIME, F\_ AMOUNT, F\_ INCOME , B\_ C\_ ID)

Based on the DDL file of the database and the prepared data, complete the following functions:

1. A new customer, whose ID number is "610103123456781234", registers at bank C, and applys for a new debit card. Please insert the record of the customer in the customer table and the card table.
2. Add table constraints according to the business needs, and verify them after the constraints are added successfully.
3. In the bank card table, financial product purchase table, insurance purchase table and fund purchase table, add the correct foreign key constraints: that is, the customer number is set as a foreign key, and references the customer number in the customer table, and the financial product number, insurance number and fund number refer to the corresponding table respectively.
4. In the above basic tables, there are six attributes related to amount or price. In the real life, the amount or price will not be negative. Therefore, for these properties, add constraints whose value is greater than 0.
5. Simulate the following business to write SQL statements for query:
6. Query the card number and type information of all bank cards of bank C.
7. Query the number of customers owned by bank C.
8. query the customer number, name and ID number information of the bank cards owners.
9. Statistics the amount of debit cards and credit cards respectively in all the bank cards.
10. Query the average insurance price in the insurance table.
11. Query the insurance type and price corresponding to the maximum and minimum of insurance price in the insurance table.
12. Query the customer number and name whose bank card number is' 62220213020200006 '.
13. Query the insurance name and applicable population whose insurance price is greater than the average value in the insurance product.
14. Query the total number of financial products released by Bank C, by using the P\_ YEAR to group.
15. Query the insurance number, insurance name and insurance period applicable to the elderly people.
16. Create a business view based on the following queries:
17. Create a view containing customer number, name and ID number of all the customers who have at least one bank card.
18. Modify view: on the basis of the original view, only the customers with credit cards are included.
19. Simulate the business changes, people's demand for fund query has increased significantly. Create a composite index on the fund purchase table:

C\_ id ASC, f\_ id ASC, f\_ amount DESC

1. Simulate the business demands, add, delete and modify data:
2. A customer with customer number 2, applys to update his phone numbers with '13312345678'.
3. Simulate the simplified process of the sale, purchase, settlement and stop saling of No. 4 financial product.

* **Step 1: The bank sales new financial product.**

On December 1, 2018, the bank will sell one-year financial product, No. 4, starting closing time is 2019 / 1 / 6, price is 8.0 yuan per share, status of 0 (0 means normal).

Hint: it is required to insert the appropriate record into table finances\_product.

* **Step 2: Customers purchase No. 4 financial products (purchase and deduct money from corresponding bank card).**

Suppose there are three deals:

a. At 13:00:00:00 on January 5, 2019: customer 3 purchased 1000 shares of No. 4 financial product with bank card '62220213020000002':

b. At 14:00:00:00 on January 5, 2019: customer 5 purchased 1000 shares of No. 4 financial product with bank card '62222021302020200003'

c. 2015 / 1 / 5 15:00:00: customer 5 purchased 500 shares of financial product No.4 with bank card '62220213020000003'

Hint: you need to give insert some appropriate purchase records into the table C\_finances, and the total purchase amount is calculated according to the purchase amount and price(unit price), and the corresponding amount is deducted from the purchased bank card. Pay attention to consider the occasion of insufficient balance of debit card, which may lead to purchase failure. The whole process involves multiple tables, so you can consider using procedures to encapsulate the whole process. In addition, the balance of the bank card should be updated， if the balance of the debit card is insufficien, the update would be failed.

* Step 3: The bank cashes the income

Assuming that the financial product expired on January 6, 2020, the income is assumed to be 5%. On January 6, 2020, the bank will cash in the proceeds for all users who purchase No. 4 financial product: calculate the income of all customers who purchased the product, and add the total amount of principal + income to the bank card balance.

Hint: according to the specified rate of income, calculate the balance (income + principal) of each purchase transaction of No. 4 financial product. The balance of the transaction is then added to the balance of the payment card of the transaction.

* Step 4: The bank stops the financial product No.4

It is assumed that the bank has finished processing the income cashing of financial product No.4. Now the bank wants to take the financial product off the shelf. Note that the foreign key constraints should be considered when deleting the financial product record.

Hint: you’d better backup the data before delete. For example, save the historical purchase records of No. 4 financial product to a separate history table, and then delete the relevant data of No.4 financial product from table C\_ finances. When delete No.4 financial product from table finances\_product, the foreign key constraint should be considered, and the purchase record of product 4 should be deleted first. We can also consider adding a column to finances\_ product, to make the product property be invalid instead of deleting it completely.

1. Use JDBC, ODBC or other third-party library programming to implement the connection to the finance database, and implement the function of step 2 and step 3 in question (6) above.

# Experiment 7：Transaction and Concurrency Control

## Goal

To practice the

1. To grasp the concept transaction, and how to create the transaction.
2. Understand the data inconsistency problems in concurrency operation, and can use lock and ioslation mechanisms

## Content

Suppose that the university allows students to bind the bank card with the campus card. There are the following basic tables in the student database, in which the campus card number (cardid) is the student number:

**icbc\_card(studcardid, icbcid, balance) //campus ID，icbc bank ID，balance of bank card**

**campus\_card(studcardid, balance) // campus card ID，c\_card balance**

**Some example data in this experiment**

**create table icbc\_card(**

**icbcid int,**

**sno varchar(8),**

**balance int**

**);**

**create table campus\_card(**

**sno varchar(8),**

**balance int**

**);**

**insert into campus\_card values ('20200032', 1);**

**insert into campus\_card values ('20200033', 100);**

**insert into icbc\_card values (1, '20200032', 300);**

**insert into icbc\_card values (2, '20200033', 400);**

According to the requirements, try to complete the following experiments based on the database above:

1. Write a transaction to achieve the following operations: a student( student number is 20200032) transfers 200 yuan from the bank card to the campus card, and if there is a failure during the transfer process, it will be rolled back. (10 points)
2. According to the database and tables, use specific examples to show several data inconsistency problems: such as missing and modifying, reading dirty data, non repeatable reading and phantom reading (deletion and insertion). If there is any situation that cannot be displayed, please explain the reasons. (20 points, 10 points for each data inconsistency)
3. By using the isolation levels or lock mechanism of the database, design solutions to solve the data inconsistency problems you have set in question 2. (20 points, 5 points for each data inconsistency)
4. Construct two transactions and update one tuple in the database at the same time. Try to use the following SQL commands to view and understand the feed back information of transaction and lock status in the current system. (10 points)

* show engine innodb status (MySQL 8.0 or 5.7)
* select \* from information\_schema.innodb\_trx (MySQL 8.0 or 5.7)
* select \* from performance\_schema.data\_locks; (MySQL 8.0)
* select \* from sys.innodb\_lock\_waits; (MySQL 8.0)
* select \* from information\_schema.innodb\_lock\_waits (MySQL 5.7)
* select \* from information\_schema.innodb\_locks (MySQL 5.7)

1. Construct a deadlock situation. (10 points)
2. Construct the transaction containing some ‘savepoint’ and roll back to a savepoint at a certain time. (10 points)
3. Through experiments try to check all kinds of logs in MySQL: query log, error log and slow query log. (10 points)
4. Use mysqlbinlog to view the transaction log of the database, and try to recover the data according to the following scenarios. (10 points)

Steps：

1) crate db1, and create table t1, t2 . Table t1,t2 structures are the same: create table t1 (id int);

2) Insert data into t1: 11,12,13

3) Insert data into t2: 21, 22, 23

3) drop table t1;

4) t2 can also be used normally, insert data 24 into t2

After finished the 4th step above, use mysqlbinlog to recover T1 data.

The SQL of the above four steps is as follows:

create database db1;

create table t1(id int);

create table t2(id int);

insert into t1 values(11);

insert into t1 values(12);

insert into t1 values(13);

insert into t2 values(21);

insert into t2 values(22);

insert into t2 values(23);

drop table t1;

insert into t2 values(24);

# Experiment 8：Project

## Goal

The goal of this project is to provide a realistic experience in the conceptual design, logical design, implementation, operation, and maintenance of a relational database and associated applications.

## Requirement

1. Experimental steps

(1) First of all, do a simple requirement analysis for the selected experiment.

(2) Make data flow diagram and data dictionary.

(3) Design E-R diagram (conceptual structure design) based on the data flow diagram and dictionary.

(4)Based on the E-R diagram, design the relational schema (at least meet 3NF) (logical structure design).

(5) According to various data requests in requirement analysis, obtain various views and descriptions of constraints, rules and trigger scripts.

(6) According to the content of (5), design the database in mysql, or SQL server, etc.

(7) To realize the designed functions with one language you are familiar with. The design must include the query, addition, modification and deletion of data.

(8)Write a document for the previous steps, collect program source code, executable program.

(9) At the end of the experiment report, write the problems you have encountered in the experiment and the solutions.

1. Submission of achievements:
2. Complete experimental report (the report includes( 1)-(5 )and (9 )in the above requirements). Note that it is not necessary to paste the source code of client program into the report, but the code of database such as trigger and stored procedure can be described and written into the experiment report. In addition, please write the development environment (including language, platform, name and version of the database you used) into the experiment report.
3. Database backup file, as well as related SQL statements.
4. Source code and executable program.
5. Configuration document: describes how to configure the program to run normally.
6. Other: installation package, help documents, demo video, etc. (optional).
7. Marking criteria:

Requirement analysis: 5 points (data flow diagram and data dictionary)

Concept design: 10 points (E-R diagram)

Logical design: 10 points (relational schema, including function dependences, primary key, etc. optimization, table design)

Physical design: 5 points (index design, etc.)

Program source code: 60 points

Others: 10 points (document, environment configuration, etc.)

## Experiment systems

Choose one of the following topics or choose your own topic for experiment, and write the experiment report. The functions listed in each of the following topics are for reference only. You can design the functions by yourself in the experiment.

### Library Management System

A simple library management system includes the information of books in the library, the information of students and the borrowing information of students. The function of the system is divided into two parts: student oriented and administrator oriented. The student oriented part contains borrow, renew, return and query books; the administrator oriented part contains complete the addition, deletion and modification of books and students, as well as the confirmation of students' borrowing, renewing and returning.

### Student Status Management System

Student status management system can mainly manage students' files and grades.

1. Establish student files, you can input, modify and query the student status information.

2. You can input, modify and query the student's grades and average scores.

### Ticket Selling Management System

Investigate the city's long-distance bus station or railway station ticket business, design the ticket saling management system. The following functions are required:

1. It has convenient and fast ticket selling function, including ticket reservation and refund function, supporting group reservation and refund.
2. It is convenient to query the timetable or train number, ticket price and other information.
3. Various information stored in the system can be changed.

### Enterprise personnel management system

To understand the personnel management of the enterprise, and design enterprise personnel management system according to the specific situation. The main functions are as follows:

1. Recruitment and employment of employees.
2. Attendance, overtime, business trip management.
3. Personnel changes: new employee registration, employee resignation registration, etc.
4. Employee training records.
5. Employee assessment, reward and punishment records.

### Telephone payment management system

According to the current telecom, China Mobile, Unicom and other specific circumstances, design the telephone payment management system. The main functions are as follows:

1. Registration and management of telephone user information.
2. Payment of telephone charges.
3. Modification of various telephone charges.
4. Various inquiries about telephone charges.
5. Various statistical functions.

# Appendix

## mysql common commands

1. Database

* Connection to database: MySQL - H 192.168.10.89 - U root - P
* View database: show databases
* Using database: use [database name]
* Check the database creation SQL: show create database [database name];

2. Tables

* View table list: show tables;
* Check the table creation SQL: show create table [table name];
* Check the index: Show index from [table name];

3. Users

* Check the current user: select user();
* Check the list of MySQL permission Keywords: Show privilege
* Check the permissions of a user: Show grants for ‘root’@ ‘%’;

## Data generation

Generate random numbers between [0,10000]: select rand() \* 10000;

Generate random integers between [0,10000]: select floor (rand() \* 10000);

Generate random integers between [50,100]: select floor (rand() \* (100-50) + 50);

Generate random string: select substring (md5 (rand()), 1,10);

Generate random dates between [2017 / 01 / 01-2018 / 01 / 01]:

select date(from\_ unixtime( unix\_ timestamp('2017-01-01') +

floor( rand() \* (unix\_ timestamp('2018-01-01')-unix\_ timestamp('2017-01-01') + 1 ) ))) as date;

## Time monitoring

Function to calculate time difference: TIMESTAMPDIFF（MINUTE, start\_time,end\_time）