

数据库系统课程上机实验手册

华为 GaussDB 版

大连理工大学

Dalian University of Technology

目录

第一章 实验介绍.....	3
1.1 关于本实验.....	3
1.2 实验目的.....	3
1.3 内容描述.....	3
第二章 DDL.....	3
2.1 预备知识.....	3
2.2 实验任务.....	4
2.3 SQL 代码与对应结果.....	4
第三章 DML.....	6
3.1 预备知识.....	6
3.2 实验任务.....	6
第四章 数据查询.....	10
4.1 单表查询.....	10
4.2 聚合查询.....	14
4.3 多表查询.....	15
4.4 子查询.....	17
4.5 集合查询.....	19
第五章 索引操作.....	20
5.1 预备知识.....	20
5.2 实验任务.....	21
第六章 事务的并发控制.....	22
6.1 预备知识.....	22
6.2 实验任务.....	22

第一章 实验介绍

1.1 关于本实验

本实验共包含 5 个子实验,从数据准备开始,逐一介绍了 scoot 数据库中的 SQL 语法,包括数据查询、数据更新、数据定义和数据控制。

1.2 实验目的

学习 SQL 相关操作,了解 GaussDB 系统对索引和事务并发控制的实现。

1.3 内容描述

子实验 2 为数据定义实验,介绍了 DDL 的类型、语法格式和使用场景,帮助读者熟练掌握如何用数据定义语言定义或修改数据库中的对象。

子实验 3 为数据更新实验,通过对 DML 语言基本语法和使用,帮助读者掌握如何对数据库中数据进行更新操作,包括数据插入、数据修改和数据删除。

子实验 4 为数据查询实验,通过基本的 DQL 语言使用,帮助读者掌握从一个或多个表查询数据的操作。

子实验 5 为索引操作,通过基本的索引使用,帮助读者掌握索引的创建和使用。

子实验 6 为事务并发控制,通过对事务并发场景的设计,帮助读者了解 GaussDB 数据库关系对并发的支持情况。

第二章 DDL

2.1 预备知识

数据库模式定义语言 DDL(Data Definition Language),是用于描述数据库中要存储的现实世界实体的语言,用来创建数据库中的各种对象-----表、视图、索引、同义词、聚簇等

```
create table 表名(
```

字段名 1 类型[(宽度) 约束条件],
字段名 2 类型[(宽度) 约束条件],
字段名 3 类型[(宽度) 约束条件]
);
#类型：使用限制字段必须以什么样的数据类型传值
#约束条件：约束条件是在类型之外添加一种额外的限制

2.2 实验任务

创建 DEPT、BONUS、SALGRADE、EMP 表，关系模式为：
DEPT (DEPTNO INT, DNAME VARCHAR(14), LOC VARCHAR(13));
EMP (EMPNO INT, ENAME VARCHAR(10), JOB VARCHAR(9), MGR INT, HIREDATE
DATE, SAL FLOAT, COMM FLOAT, DEPTNO INT);
BONUS (ENAME VARCHAR(10), JOB VARCHAR(9), SAL INT, COMM INT);
SALGRADE (GRADE INT, LOSAL INT, HISAL INT);
注意：EMP 表中的 DEPTNO 属性为外键，对应 DEPT 表中的 DEPTNO

2.3 SQL 代码与对应结果

```
CREATE TABLE DEPT (  
DEPTNO    INT,  
DNAME     VARCHAR(14),  
LOC       VARCHAR(13),  
CONSTRAINT PK_DEPT PRIMARY KEY (DEPTNO)  
);
```

```
CREATE TABLE BONUS (  
ENAME     VARCHAR(10),  
JOB       VARCHAR(9),  
SAL       INT,  
COMM      INT  
);
```

```
CREATE TABLE SALGRADE (  
GRADE     INT,
```

```
LOSAL      INT,  
HISAL      INT  
);
```

```
CREATE TABLE EMP (  
EMPNO      INT,  
ENAME      VARCHAR(10),  
JOB        VARCHAR(9),  
MGR        INT,  
HIREDATE   DATETIME,  
SAL        FLOAT,  
COMM       FLOAT,  
DEPTNO     INT,  
CONSTRAINT PK_EMP PRIMARY KEY (EMPNO),  
CONSTRAINT FK_DEPTNO FOREIGN KEY (DEPTNO) REFERENCES DEPT(DEPTNO)  
);
```



第三章 DML

3.1 预备知识

DML 是 Data Manipulation Language 的缩写，意思是数据操纵语言，是指在 SQL 语言中，负责对数据库对象运行数据访问工作的指令集，以 INSERT、UPDATE、DELETE 三种指令为核心，分别代表插入、更新与删除，是开发以数据为中心的应用程序必定会使用到的指令。

1 增

```
INSERT INTO tb_name (col_name1, col_name2) VALUES ('val1', 'val1'),('val2', 'val2');
```

2 删

```
DELETE FROM table_name [WHERE];
```

3 改

```
UPDATE tb_name SET col_name1 = 'new_val1', col_name2 = 'new_val2' [WHERE];
```

3.2 实验任务

3.2.1 实践题目 1

在 DEPT 表中插入数据 10, 'ACCOUNTING', 'NEW YORK'。

SQL 代码、对应结果

```
INSERT INTO DEPT VALUES(10, 'ACCOUNTING', 'NEW YORK');
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK

3.2.2 实践题目 2

在 DEPT 表中根据 DEPTNO 修改 LOC 为 BEIJING。

SQL 代码、对应结果

```
UPDATE dept SET LOC = 'BEIJING' WHERE DEPTNO=10;
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	BEIJING

3.2.3 实践题目 3

在 DEPT 表中删除 DEPTNO 为 10 的数据。

SQL 代码、对应结果

```
DELETE FROM dept WHERE DEPTNO=10;
```

DEPTNO	DNAME	LOC
(Null)	(Null)	(Null)

3.2.4 实践题目 4

在 DEPT 表中插入两条数据 10, 'ACCOUNTING', 'NEW YORK' 和 20, 'RESEARCH', 'DALLAS'。

SQL 代码、对应结果

```
INSERT INTO DEPT VALUES(10, 'ACCOUNTING', 'NEW YORK');
```

```
INSERT INTO DEPT VALUES(20, 'RESEARCH', 'DALLAS');
```

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS

3.2.5 实践题目 5

删除 DEPT 表中所有数据。

SQL 代码、对应结果

```
DELETE FROM dept;
```

DEPTNO	DNAME	LOC
(Null)	(Null)	(Null)

3.2.6 实践题目 6

根据下表为数据库中插入数据。

DEPT表：

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

EMP表:

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7369	SMITH	CLERK	7566	1980-12-17 00:00:00.000	800	NULL	20
2	7499	ALLEN	SALESMAN	7698	1981-02-20 00:00:00.000	1600	300	30
3	7521	WARD	SALESMAN	7698	1981-02-22 00:00:00.000	1250	500	30
4	7566	JONES	MANAGER	7839	1981-04-02 00:00:00.000	2975	NULL	20
5	7654	MARTIN	SALESMAN	7698	1981-09-28 00:00:00.000	1250	1400	30
6	7698	BLAKE	MANAGER	7839	1981-05-01 00:00:00.000	2850	NULL	30
7	7782	CLARK	MANAGER	7839	1981-06-09 00:00:00.000	2450	NULL	10
8	7788	SCOTT	ANALYST	7566	1987-06-13 00:00:00.000	3000	NULL	20
9	7839	KING	PRESIDENT	NULL	1981-11-17 00:00:00.000	5000	NULL	10
10	7844	TURN...	SALESMAN	7698	1981-09-08 00:00:00.000	1500	0	30
11	7876	ADAMS	CLERK	7788	1987-06-13 00:00:00.000	1100	NULL	20
12	7900	JAMES	CLERK	7698	1981-12-03 00:00:00.000	950	NULL	30
13	7934	MILLER	CLERK	7782	1982-01-23 00:00:00.000	1300	NULL	10

SALGRADE表:

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999

BONUS表: 无数据

SQL 代码

```
INSERT INTO DEPT VALUES
(10, 'ACCOUNTING', 'NEW YORK');
INSERT INTO DEPT VALUES
(20, 'RESEARCH', 'DALLAS');
INSERT INTO DEPT VALUES
(30, 'SALES', 'CHICAGO');
INSERT INTO DEPT VALUES
(40, 'OPERATIONS', 'BOSTON');
```

```
INSERT INTO EMP VALUES
(7369, 'SMITH', 'CLERK', 7566, '1980-12-17', 800, NULL, 20);
INSERT INTO EMP VALUES
(7499, 'ALLEN', 'SALESMAN', 7698, '1981-02-20', 1600, 300, 30);
```



```

INSERT INTO EMP VALUES
(7521, 'WARD', 'SALESMAN', 7698, '1981-02-22', 1250, 500, 30);
INSERT INTO EMP VALUES
(7566, 'JONES', 'MANAGER', 7839, '1981-04-02', 2975, NULL, 20);
INSERT INTO EMP VALUES
(7654, 'MARTIN', 'SALESMAN', 7698, '1981-09-28', 1250, 1400, 30);
INSERT INTO EMP VALUES
(7698, 'BLAKE', 'MANAGER', 7839, '1981-05-01', 2850, NULL, 30);
INSERT INTO EMP VALUES
(7782, 'CLARK', 'MANAGER', 7839, '1981-06-09', 2450, NULL, 10);
INSERT INTO EMP VALUES
(7788, 'SCOTT', 'ANALYST', 7566, '1987-06-13', 3000, NULL, 20);
INSERT INTO EMP VALUES
(7839, 'KING', 'PRESIDENT', NULL, '1981-11-17', 5000, NULL, 10);
INSERT INTO EMP VALUES
(7844, 'TURN...', 'SALESMAN', 7698, '1981-09-08', 1500, 0, 30);
INSERT INTO EMP VALUES
(7876, 'ADAMS', 'CLERK', 7788, '1987-06-13', 1100, NULL, 20);
INSERT INTO EMP VALUES
(7900, 'JAMES', 'CLERK', 7698, '1981-12-03', 950, NULL, 30);
INSERT INTO EMP VALUES
(7934, 'MILLER', 'CLERK', 7782, '1982-01-23', 1300, NULL, 10);

```

```

INSERT INTO SALGRADE VALUES
(1, 700, 1200);
INSERT INTO SALGRADE VALUES
(2, 1201, 1400);
INSERT INTO SALGRADE VALUES
(3, 1401, 2000);
INSERT INTO SALGRADE VALUES
(4, 2001, 3000);
INSERT INTO SALGRADE VALUES
(5, 3001, 9999);

```

第四章 数据查询

4.1 单表查询

4.1.1 预备知识

单表查询是最简单的查询方式。所有要查询的信息，都集中在一张表中。也就是说，SQL 语句中的 FROM 子句中只有一个表。我们可以通过以下的几道实践题目来巩固这个知识点。

4.1.2 实践题目 1

查看 EMP 表中部门号为 10 的员工的姓名，职位，参加工作时间，工资。

结果：

	ename	job	hiredate	sal
1	CLARK	MANAGER	1981-06-09 00:00:00.000	2450
2	KING	PRESIDENT	1981-11-17 00:00:00.000	5000
3	MILLER	CLERK	1982-01-23 00:00:00.000	1300

4.1.3 实践题目 2

查所有已有的职位，要求去除重复项。

结果：

	job
1	ANALYST
2	CLERK
3	MANAGER
4	PRESIDENT
5	SALESMAN

4.1.4 实践题目 3

计算每个员工的年薪，并取列名为 Salary of Year (emp.sal 为员工的月薪)，要求输出员工姓名，年薪。

结果：

	ename	Salary Of Year
1	SMITH	9600
2	ALLEN	19200
3	WARD	15000
4	JONES	35700
5	MARTIN	15000
6	BLAKE	34200
7	CLARK	29400
8	SCOTT	36000
9	KING	60000
10	TURNER	18000
11	ADAMS	13200
12	JAMES	11400
13	MILLER	15600

4.1.5 实践题目 4

查询每个员工每个月拿到的总金额（emp.sal 为工资，emp.comm 为补助）。（提示：gaussdb 中，nvl (ex1,ex2) 表示如果 ex1 为空则返回 ex2）

结果：

	ename	total
1	SMITH	800
2	ALLEN	1900
3	WARD	1750
4	JONES	2975
5	MARTIN	2650
6	BLAKE	2850
7	CLARK	2450
8	SCOTT	3000
9	KING	5000
10	TURNER	1500
11	ADAMS	1100
12	JAMES	950
13	MILLER	1300

4.1.6 实践题目 5

显示职位是主管（manager）的员工的姓名，工资。

结果：

	ename	sal
1	JONES	2975
2	BLAKE	2850
3	CLARK	2450

4.1.7 实践题目 6

显示第 3 个字符为大写 O 的所有员工的姓名及工资。

结果：

	ename	sal
1	SCOTT	3000

4.1.8 实践题目 7

显示职位为销售员 (SALESMAN) 或主管 (MANAGER) 的员工的姓名，工资，职位。

结果：

	ename	sal	job
1	ALLEN	1600	SALESMAN
2	WARD	1250	SALESMAN
3	JONES	2975	MANAGER
4	MARTIN	1250	SALESMAN
5	BLAKE	2850	MANAGER
6	CLARK	2450	MANAGER
7	TURNER	1500	SALESMAN

4.1.9 实践题目 8

显示所有没有补助的员工的姓名。

结果：

	ename
1	SMITH
2	JONES
3	BLAKE
4	CLARK
5	SCOTT
6	KING
7	ADAMS
8	JAMES
9	MILLER

4.1.10 实践题目 9

显示有补助的员工的姓名，工资，补助。

结果：

	ename	sal	comm
1	ALLEN	1600	300
2	WARD	1250	500
3	MARTIN	1250	1400
4	TURNER	1500	0

4.1.11 实践题目 10

排序显示所有员工的姓名，工资（按工资降序方式）。

结果：

	ename	sal
1	KING	5000
2	SCOTT	3000
3	JONES	2975
4	BLAKE	2850
5	CLARK	2450
6	ALLEN	1600
7	TURNER	1500
8	MILLER	1300
9	WARD	1250
10	MARTIN	1250
11	ADAMS	1100
12	JAMES	950
13	SMITH	800

4.1.12 实践题目 11

显示员工的最高工资和最低工资。

结果：

	最高工资	最低工资
1	5000	800

4.1.13 实践题目 12

显示所有员工的平均工资和总计工资。

结果：

	平均工资	总计工资
1	2001.92307692308	26025

4.1.14 实践题目 13

显示补助在员工中的发放比例、即有多少比例的员工有补助。（此题需注意两个问题：1.select 语句中进行除法如何保留小数点后数据。2.count 函数如何处理 null 型数据。）

结果：

	(无列名)
1	0.307692307692

4.2 聚合查询

4.2.1 预备知识

在查询中，我们经常会遇到这样的问题：求平均值、求最值等等。我们需要使用一些函数如 AVG(), MAX()等来进行计算，也需要通过 GROUP BY 子句来聚合属性。

4.2.2 实践题目 1

显示每种职业的平均工资。

结果：

	job	average
1	ANALYST	3000
2	CLERK	1037.5
3	MANAGER	2758.333333333333
4	PRESIDENT	5000
5	SALESMAN	1400

4.2.3 实践题目 2

显示每个部门每种岗位的平均工资和最高工资。

结果：

	deptno	job	average	max
1	20	ANALYST	3000	3000
2	10	CLERK	1300	1300
3	20	CLERK	950	1100
4	30	CLERK	950	950
5	10	MANAGER	2450	2450
6	20	MANAGER	2975	2975
7	30	MANAGER	2850	2850
8	10	PRESIDENT	5000	5000
9	30	SALESMAN	1400	1600

4.2.4 实践题目 3

显示平均工资低于 2500 的部门号，平均工资及最高工资。

结果：

	deptno	average	max
1	20	1968.75	3000
2	30	1566.66666666667	2850

4.2.5 实践题目 4

上一条语句以平均工资升序排序。

结果：

	deptno	average	max
1	30	1566.66666666667	2850
2	20	1968.75	3000

4.3 多表查询

4.3.1 预备知识

在大部分情况下，我们所需要的信息并不仅仅包含在一张表中。我们首先需要使用 join 连接多个表，然后再进行查询

4.3.2 实践题目 1

显示工资高于 2500 或岗位为 MANAGER 的所有员工的姓名，工资，职位，和部门号。

结果：

	ename	sal	job	deptno
1	JONES	2975	MANAGER	20
2	BLAKE	2850	MANAGER	30
3	CLARK	2450	MANAGER	10
4	SCOTT	3000	ANALYST	20
5	KING	5000	PRESIDENT	10

4.3.3 实践题目 2

排序显示所有员工的姓名，部门号，工资（以部门号升序，工资降序，雇用日期升序显示）。

结果：

	ename	deptno	sal
1	KING	10	5000
2	CLARK	10	2450
3	MILLER	10	1300
4	SCOTT	20	3000
5	JONES	20	2975
6	ADAMS	20	1100
7	SMITH	20	800
8	BLAKE	30	2850
9	ALLEN	30	1600
10	TURNER	30	1500
11	WARD	30	1250
12	MARTIN	30	1250
13	JAMES	30	950

4.3.4 实践题目 3

采用自然连接的原理显示部门名以及相应的员工姓名。（Sql server 不支持 NATURAL JOIN 语法。）

结果：

	dname	ename
1	RESEARCH	SMITH
2	SALES	ALLEN
3	SALES	WARD
4	RESEARCH	JONES
5	SALES	MARTIN
6	SALES	BLAKE
7	ACCOUNTING	CLARK
8	RESEARCH	SCOTT
9	ACCOUNTING	KING
10	SALES	TURNER
11	RESEARCH	ADAMS
12	SALES	JAMES
13	ACCOUNTING	MILLER

4.3.5 实践题目 4

查询 SCOTT 的上级领导的姓名。

结果：

	ename
1	JONES

4.3.6 实践题目 5

显示部门的部门名称，员工名即使部门没有员工也显示部门名称。

结果：

	dname	ename
1	ACCOUNTING	CLARK
2	ACCOUNTING	KING
3	ACCOUNTING	MILLER
4	RESEARCH	SMITH
5	RESEARCH	JONES
6	RESEARCH	SCOTT
7	RESEARCH	ADAMS
8	SALES	ALLEN
9	SALES	WARD
10	SALES	MARTIN
11	SALES	BLAKE
12	SALES	TURNER
13	SALES	JAMES
14	OPERATIONS	NULL

4.4 子查询

4.4.1 预备知识

子查询就是指的在一个完整的查询语句之中，嵌套若干个不同功能的小查询，从而一起完成复杂查询的一种编写形式。

4.4.2 实践题目 1

显示所有员工的名称、工资以及工资级别。

结果：

	ename	sal	grade
1	SMITH	800	1
2	ADAMS	1100	1
3	JAMES	950	1
4	WARD	1250	2
5	MARTIN	1250	2
6	MILLER	1300	2
7	ALLEN	1600	3
8	TURNER	1500	3
9	JONES	2975	4
10	BLAKE	2850	4
11	CLARK	2450	4
12	SCOTT	3000	4
13	KING	5000	5

4.4.3 实践题目 2

显示 ACCOUNTING 部门所有员工的名称，工资。

结果：

	ename	sal
1	CLARK	2450
2	KING	5000
3	MILLER	1300

4.4.4 实践题目 3

显示职位属于 10 号部门所提供职位范围的员工的姓名，职位，工资，部门号。

结果：

	ename	job	sal	deptno
1	SMITH	CLERK	800	20
2	JONES	MANAGER	2975	20
3	BLAKE	MANAGER	2850	30
4	CLARK	MANAGER	2450	10
5	KING	PRESIDENT	5000	10
6	ADAMS	CLERK	1100	20
7	JAMES	CLERK	950	30
8	MILLER	CLERK	1300	10

4.4.5 实践题目 4

显示工资比 30 号部门中所有员工的工资都高的员工的姓名，工资和部门号。

结果：

	ename	sal	deptno
1	JONES	2975	20
2	SCOTT	3000	20
3	KING	5000	10

4.5 集合查询

4.5.1 预备知识

当两张表的属性相同时，我们可以对它们做一些集合运算，如并集、交集等。

4.5.2 实践题目 1

显示工资高于 2500 或职位为 MANAGER 的员工的姓名，工资和职位（采用 UNION 语法实现）。

结果：

	ename	sal	job
1	BLAKE	2850	MANAGER
2	CLARK	2450	MANAGER
3	JONES	2975	MANAGER
4	KING	5000	PRESIDENT
5	SCOTT	3000	ANALYST

4.5.3 实践题目 2

显示工资高于 2500 且职位为 MANAGER 的员工的姓名，工资和职位（采用 INTERSECT 语法实现）。

结果：

	ename	sal	job
1	BLAKE	2850	MANAGER
2	JONES	2975	MANAGER

4.5.4 实践题目 3

显示工资高于 2500 但职位不是 MANAGER 的员工的姓名，工资和职位（采用 MINUS 语法实现）。

结果：

	ename	sal	job
1	KING	5000	PRESIDENT
2	SCOTT	3000	ANALYST

第五章 索引操作

5.1 预备知识

索引可以提高数据的访问速度，但同时也增加了插入、更新和删除表的处理时间。所以是否要为表增加索引，索引建立在哪些字段上，是创建索引前必须要考虑的问题。需要分析应用程序的业务处理、数据使用、经常被用作查询条件或者被要求排序的字段来确定是否建立索引。索引相关的 DDL 包括创建索引、删除索引属性和删除索引。

利用索引检索，需要在大数据量的情况下才能够体现出效率差距。因此，本实验需要利用 PLSQL（过程化 SQL）知识自动构建一个含有大量数据的表作为实验对象。PLSQL 相关知识请参考“HCIP-GaussDB-OLTP V1.0 培训教材”第 236 页



- PLSQL是一种高级数据库程序设计语言。
- 全称：
 - 过程化SQL语言 (Procedure Language & Structured Query Language)。
- 定义：
 - 一组为了完成特定功能的SQL语句的集合。

第236页 版权所有 © 2019 华为技术有限公司



下面给出实验中可能用到的 PLSQL 代码示例，该代码用于向某个表循环插入 100 个数据。其中 count 是一个变量

```
declare
count int;
begin
count := 1;
while count<=100 loop
insert into table_name(id, name, password)
values (count, 'some_name', 'some_name' || count);
count := count + 1;
end loop;
end;
```

5.2 实验任务

创建一个表，然后向该表中循环插入大量数据（1000000 条以上）。然后，对于该表中某个属性建立索引。然后，进行对比实验，对比利用索引和不利用索引的情况下，完成相同查询任务花费的时间，验证利用索引查询是否真的可以提升查询效率。

注：给出每个步骤的 **SQL** 语句，并对语句执行结果进行截图说明，尤其是其查询时间的区别截图。

第六章 事务的并发控制

6.1 预备知识

事务并发可能带来的 3 大类问题：

- 1) dirty read(脏读)，一个事务读取了另外一个事务未提交的数据。
- 2) non-repeatable read(不可重复读)，同一事务中，前后读取的数据不一致。
- 3) phantom read(幻读)，类似不可重复读，但针对插入/删除操作。

美国国家标准协会在 SQL 标准中，对于满足事务的隔离性的程度划分为以下四个级别。未提交读、已提交读、可重复读和序列化。不同的隔离级别能够解决的上述问题也不同（参考课程第一节）

6.2 实验任务

通过并发实验设计，验证 GaussDB 是否存在上述三类问题。根据验证结果给出 GaussDB 系统默认的事务隔离级别。

注：实验设计可以模仿 10.6 节中的操作，设计两个事务的并发操作场景，判断是否存在对应问题。需要详细的实验设计过程描述和实验结果的截图。