广州大学学生实验报告

开课实验室： 电子楼 418　　2018 年 3 月 19 日

学院	计算机科学与教育软件学院	**年级、专业、班**	软件 171	**姓名**	谢金宏	**学号**	1706300001
实验课程名称	数据库原理					**成绩**	
实验项目名称	实验一 数据库及数据表的创建与删除					**指导老师**	张少宏

教师评语：

一、实验目的

掌握利用 Oracle Database Configuration Assistant 工具来创建和删除 Oracle 数据库，掌握 Oracle 中的用 Create 命令定义表的方法，以及表的完整性定义，并掌握 Oracle 中的用 Alter 命令和 Drop 命令对表的修改和删除。

二、实验环境

1. Oracle 11g 数据库软件
2. SQL Developer 客户端软件。

三、实验内容

1. 使用 Oracle Database Assistant 创建数据库。
2. 使用 Oracle Database Assistant 删除刚刚创建的数据库。
3. 使用 SQL Developer，以系统管理员身份连接到数据库实例。
4. 创建用户并赋予权限，随后以新用户的身份连接到数据库。
5. 创建以下数据表（Student（主码为 SNO）、Course（主码为 CNO）、SC（主码为（SNO、CNO）），其中 SNO 引用 Student 的 SNO 属性，CNO 引用 Course 的 CNO 属性）。
6. 向三个表格中插入 3 条数据，数据内容自编。
7. 修改 Student 表格，用 SQL 语句为 Student 表格添加一个“入学时间”属性，属性名为 Senrollment。
8. 限定 Ssex 的值只能为“男”或者“女”。
9. 修改 Course 表格，用 SQL 语句为 Course 表格添加一个“说明”属性，属性名为“Cdesc”，类型为 varchar2，长度为 200。
10. 更改 Course 表格的 Cdesc 属性，使其长度变为 500。
11. 删除刚建立的属性 Cdesc。
12. 修改 Course 表的 CPNO，使其为外码，引用 Course 表的 CNO 属性。

四、实验步骤

1. 使用 Oracle Database Assistant 创建数据库。

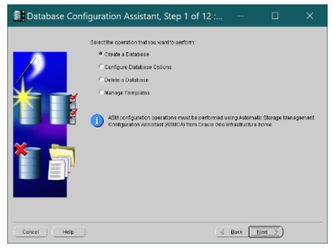


Figure 1 选择 Create a Database

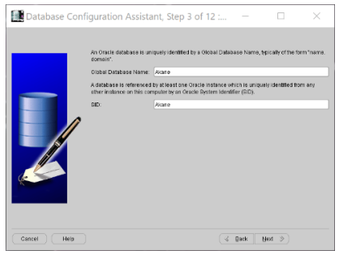


Figure 2 为数据库命名和编排 SID

启动 Oracle Database Assistant，选择 Create a Database，为数据库命名、编制 SID 和创建管理员密码。注意记忆数据的名称和 SID 以便后面连接和删除数据库的操作。

2. 使用 Oracle Database Assistant 删除刚刚创建的数据库。
启动 Oracle Database Assistant，选择 Delete a Database，选中刚刚创建的数据库删除即可。

3. 使用 SQL Developer 连接到数据库实例。
打开 SQL Developer，单击 Connection 浮动选项卡中的 New Connection..按钮，打开如下对话框。

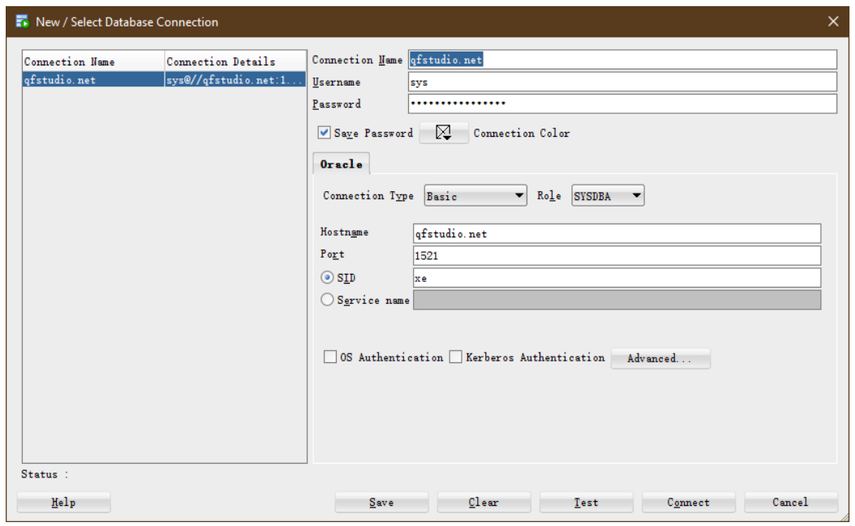


Figure 3 使用 SQL Developer 连接到数据库实例

- Connection Name 为数据库连接取的一个便于记忆的名字。
- Username 为登陆到数据库系统的用户名，系统管理员为 sys 或 system。
- Password 为用户的密码。
- Hostname 为数据库实例运行的主机名，本机为 localhost，截图中为 qfstudio.net 上的远程计算机。
- SID 为数据库 Service ID。Oracle 11g Enterprise 版本默认数据库的 SID 为 *orcl*，Express 版本默认为 *xe*。

此外，如果以 SYS 身份登陆数据库，则 Role 需要被设置为 SYSDBA，否则数据库系统会拒绝此连接。填入信息后点击 Test 可以测试数据库连接是否正常，点击 Connect 即可连接到数据库，打开 SQL 工作表。

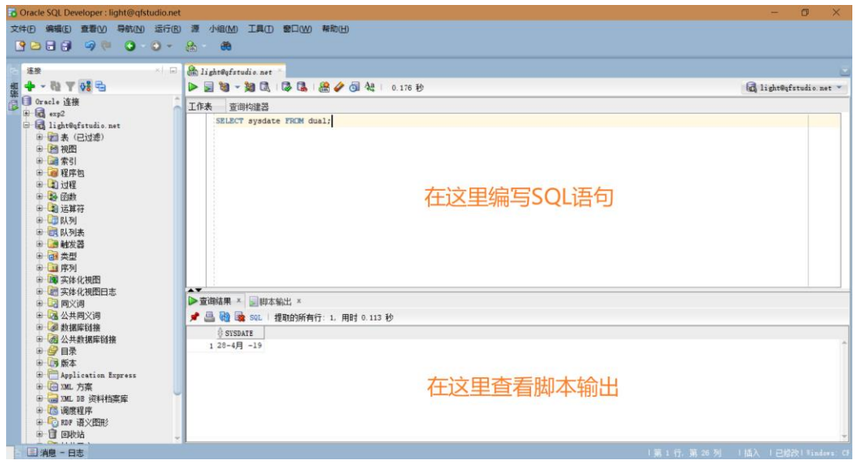


Figure 4 SQL Developer 软件截图

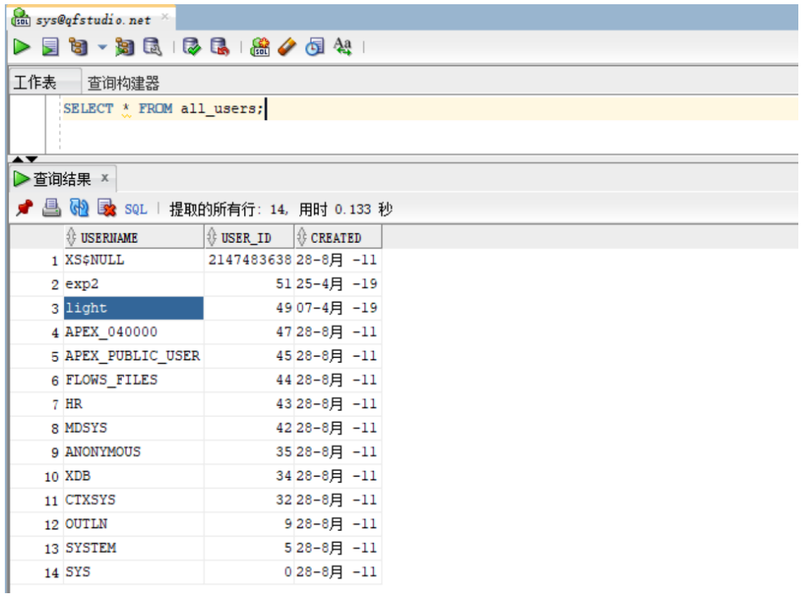
4. 创建用户并赋予权限。
以 sys 身份登陆后，使用以下语句创建新用户，并授权新用户。

```
CREATE USER light IDENTIFIED BY "lightpwd";
GRANT RESOURCE, CONNECT TO light;
```

其中 RESOURE 权限为典型的授予最终用户的权利，包括建立数据库连接、修改会话等等；CONNECT 则是典型的授予给开发人员的权利，包括建立表和修改表等等。
新用户创建成功后，改用新用户身份进行数据库连接。

使用以下 SQL 语句可以查询数据库实例中的所有用户。

```
SELECT * FROM all_users;
```



	USERNAME	USER_ID	CREATED
1	XS$NULL	2147483638	28-8月 -11
2	exp2	51	25-4月 -19
3	light	49	07-4月 -19
4	APEX_040000	47	28-8月 -11
5	APEX_PUBLIC_USER	45	28-8月 -11
6	FLOWS_FILES	44	28-8月 -11
7	HR	43	28-8月 -11
8	MDSYS	42	28-8月 -11
9	ANONYMOUS	35	28-8月 -11
10	XDB	34	28-8月 -11
11	CTXSYS	32	28-8月 -11
12	OUTLN	9	28-8月 -11
13	SYSTEM	5	28-8月 -11
14	SYS	0	28-8月 -11

Figure 5 查询系统中的用户

5. 创建以下数据表（Student（主码为 SNO）、Course（主码为 CNO）、SC（主码为（SNO、CNO）），其中 SNO 引用 Student 的 SNO 属性，CNO 引用 Course 的 CNO 属性）。

Student 表

属性名	类型	长度	是否空	含义
SNO	varchar2	10 CHAR	主码（非空）	学生编号
Sname	varchar2	5 CHAR	否	姓名
Sage	integer			年龄
Ssex	varchar2	1 CHAR		性别
Sdept	varchar2	20 CHAR		所在系

Course 表

属性名	类型	长度	是否空	含义
CNO	varchar2	5 CHAR	主码（非空）	课程编号
Cname	varchar2	10 CHAR	否	课程名
CPNO	varchar2	5 CHAR		先修课程
Ccredit	integer			学分

SC 表

属性名	类型	长度	是否空	含义
SNO	varchar2	10 CHAR	主属性（非空）	学生编号
CNO	varchar2	5 CHAR	主属性（非空）	课程编号
Grade	numeric	5,2		成绩

创建表的代码如下：

```
CREATE TABLE "Student"(
    "SNO" VARCHAR2(10 CHAR) PRIMARY KEY,
    "Sname" VARCHAR2(5 CHAR) NOT NULL,
    "Sage" INTEGER,
    "Ssex" VARCHAR2(1 CHAR),
    "Sdept" VARCHAR2(20 CHAR)
);

CREATE TABLE "Course"(
    "CNO" VARCHAR2(5 CHAR) PRIMARY KEY,
    "Cname" VARCHAR2(10 CHAR) NOT NULL,
    "CPNO" VARCHAR2(5 CHAR),
    "Ccredit" INTEGER,
    FOREIGN KEY ("CPNO") REFERENCES "Course" ("CNO")
);

CREATE TABLE "SC"(
    "SNO" VARCHAR2(10 CHAR),
    "CNO" VARCHAR2(5 CHAR),
    "Grade" NUMERIC(5, 2),
    PRIMARY KEY ("SNO", "CNO"),
    FOREIGN KEY ("SNO") REFERENCES "Student" ("SNO"),
    FOREIGN KEY ("CNO") REFERENCES "Course" ("CNO")
);
```

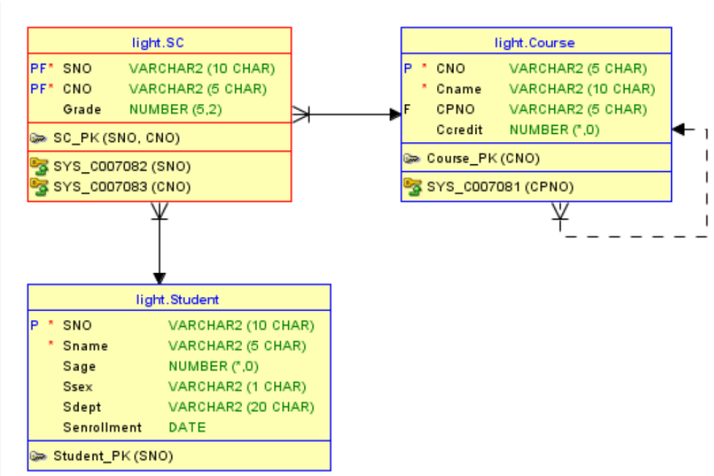


Figure 6 创建的表的模型

6. 向三个表格中插入 3 条数据，数据内容自编。

```
INSERT INTO "Course" ("CNO", "Cname", "CPNO", "Ccredit")
VALUES ('CS0', '计算机科学概论', NULL, 4);
INSERT INTO "Course" ("CNO", "Cname", "CPNO", "Ccredit")
VALUES ('CS10', '计算机图形学', 'CS0', 5);
INSERT INTO "Course" ("CNO", "Cname", "CPNO", "Ccredit")
VALUES ('CS1', 'C++', 'CS0', 5);

INSERT INTO "Student" ("SNO", "Sname", "Sage", "Ssex", "Sdept")
SELECT '1706300001', '杜新知', 21, '男', '软件工程' FROM dual UNION ALL
SELECT '1706300002', '庄梓洁', 20, '女', '计算机科学' FROM dual UNION ALL
SELECT '1706300003', '司浩轩', 22, '男', '网络工程' FROM dual;

INSERT INTO "SC" ("SNO", "CNO", "Grade")
VALUES ('1706300001', 'CS0', 100.00);
INSERT INTO "SC" ("SNO", "CNO", "Grade")
VALUES ('1706300002', 'CS0', 60.52);
INSERT INTO "SC" ("SNO", "CNO", "Grade")
VALUES ('1706300003', 'CS0', 80.899);  -- 80.899 被四舍五入为 80.90
```

7. 修改 Student 表格，用 SQL 语句为 Student 表格添加一个“入学时间”属性，属性名为 Senrollment。

```
ALTER TABLE "Student"
ADD "Senrollment" DATE DEFAULT SYSDATE;
```

8. 限定 Ssex 的值只能为“男”或者“女”。

```
ALTER TABLE "Student"
ADD CHECK("Ssex" in ('男', '女'));
```

9. 修改 Course 表格，用 SQL 语句为 Course 表格添加一个“说明”属性，属性名为“Cdesc”，类型为 varchar2，长度为 200 CHAR。

```
ALTER TABLE "Course"
ADD "Cdesc" VARCHAR2(200 CHAR);
```

10. 更改 Course 表格的 Cdesc 属性，使其长度变为 500。

```
ALTER TABLE "Course"
MODIFY "Cdesc" VARCHAR2(500 CHAR);
```

11. 删除刚建立的属性 Cdesc。

```
ALTER TABLE "Course"
DROP COLUMN "Cdesc";
```

12. 修改 Course 表的 CPNO，使其为外码，引用 Course 表的 CNO 属性。

```
ALTER TABLE "Course"
ADD FOREIGN KEY ("CPNO") REFERENCES "Course" ("CNO");
-- 报错：Self-evident
-- 因为在第 5 步创建表时已经添加了外键约束
```

从 Course 表的 DDL 中可以看出 Course 表上确有外键约束。

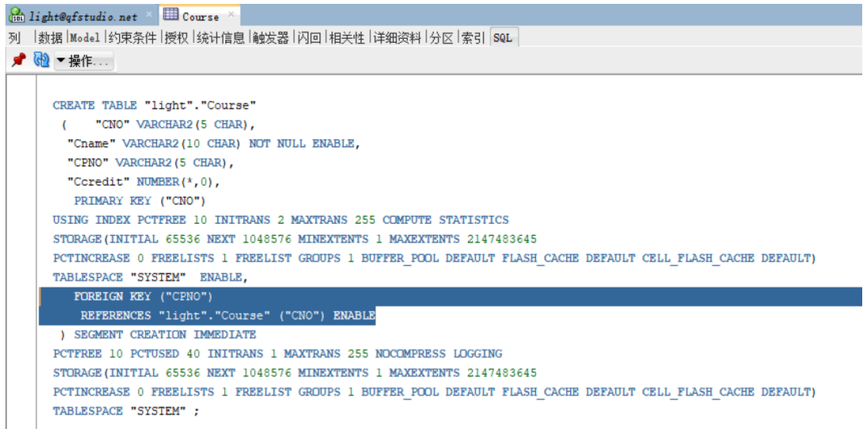


Figure 7 Course 表的 DDL

五、分析总结

本次实验的主题是数据库与数据表的创建和删除，涉及的软件有 Oracle 11G 和 SQL Developer。

Oracle 11G 是 Oracle 公司开发的关系型数据库管理软件，运行该软件的计算机可以被称为数据库服务器，简称服务器。SQL Developer 是可以连接服务器上运行的数据库管理软件的客户端，连接到服务器上之后可以通过编写和运行 SQL 语句等对数据库进行操作。

一台服务器上可以运行多个数据库实例，不同的数据库实例之间可以通过 SID 来进行区分。（注意 SID 的全程并不是 Service ID 而是 Oracle System ID。）

一个数据库即为一组特定相关联的数据的集合。数据库理论中提及的“模式 Schema”，等价于 Oracle 数据库中的用户。一个模式中可以有多个表，每一个表都代表一种关系模式，表中的列（或称“字段”）即为关系模式中的属性。其他内容此处不再赘述。

SQL 语句是操纵数据的工具。Oracle 的 SQL 实现与其他软件的实现存在一定差异，下面就我所了解的内容，尝试对 Oracle 数据库和 MariaDB 进行一些比较。

首先是标识符的差异。Oracle 对于没有使用双引号引用的标识符（如表名或列名），会默认将标识符转换成全大写的形式。Windows 上的 MariaDB 则是转换成全小写的形式（标识符敏感性与平台相关）。且 MariaDB 中标识符使用反引号“`”包围。

在第四大节的实验过程代码中使用了双引号包围标志符，以强调标识符的大小写敏感性。

在创建对象时，添加评论可能是一个好习惯，但 Oracle 实现中添加评论的方法较为复杂。

```
CREATE TABLE ORACLE
(
  COLUMN1 VARCHAR2(20)
);
COMMENT ON TABLE ORACLE IS '这是一张表';
COMMENT ON COLUMN ORACLE.COLUMN1 IS '这是一个字段';

CREATE TABLE `maria` (
    `col` INT NULL COMMENT '这是一个字段'
)
COMMENT='这是一张表';
```

向表中插入数据时，Oracle 实现中没有插入多行数据的“天然”方法，而 MariaDB 可以使用下面的语法。

```
INSERT INTO `maria` (`col1`, `col2`) VALUES
(1, 2),
(3, 4),
(5, 6);
```

在 Oracle 中插入多行数据的 workaround 方法在第四大节的第 6 小节中已经有所体现，此处不再赘述。

在 Oracle 的设计中，SELECT 查询必须依附于表，即便查询的内容是与表无关的常量（因此衍生了专为常量查询设计的 DUAL 表）。MariaDB 中 SELECT 语句可以用于输出函数的返回值，不需要依附于特定的表。以查询系统时间为例，有以下两种形式：

```
SELECT sysdate FROM dual;  -- Oracle
SELECT NOW();  -- MariaDB
```

以上为使用 Oracle 过程中发现的与其他数据库软件比较中较为显著的差异。Oracle 可能出于某种原因采用了较为繁琐的语法，也因此与“标准 SQL”较为相近。本次实验增长了我在数据库方面的见识，锻炼了适应不同环境的能力，也一定程度上让我改变了对商业软件的看法。

以上为本次实验的心得体会。