

SQL 语句的多表查询方式

例如：按照 department_id 查询 employees(员工表)和 departments(部门表)的信息。

方式一(通用型):**SELECT ... FROM ... WHERE**

```
SELECT e.last_name,e.department_id,d.department_name
FROM employees e,departments d
where e.department_id = d.department_id
```

方式二: **SELECT ... FROM ... NATURAL JOIN ...**

有局限性：会自动连接两个表中相同的列(可能有多列:department_id 和 manager_id)

```
SELECT last_name,department_id,department_name
FROM employees
NATURAL JOIN departments
```

方式三: **SELECT ... JOIN ... USING ...**

有局限性：好于方式二，但若多表的连接列列名不同，此法不合适

```
SELECT last_name,department_id,department_name
FROM employees
JOIN departments
USING(department_id)
```

方式四: **SELECT ... FROM ... JOIN ... ON ...**

常用方式，较方式一，更易实现外联接(左、右、满)

```
SELECT last_name,e.department_id,department_name
FROM employees e
JOIN departments d
ON e.department_id = d.department_id
```

--内连接

- 1)
 - 等值连接
 - 不等值连接
- 2)
 - 非自连接
 - 自连接

--外连接

--左外连接、右外连接、满外连接

创建和管理表(DDL)

CRAETE TABLE /TRUNCATE TABLE /ALTER TABLE /REANME...TO/DROP TABLE ..操作完以后，自动 commit;所以，rollback 对其操作，没有效果

1.创建表**1) 直接创建**

```
create table emp1(  
  name varchar2(20),  
  salary number(8,2)default 1000,  
  id number(4),  
  hire_date date  
);
```

2) 通过子查询的方式创建

```
create table emp2  
as  
select last_name name,employee_id id,hire_date  
from employees;
```

或者

```
create table emp2  
as  
select last_name name,employee_id id,hire_date  
from employees  
where department_id = 80;/where 1=2;
```

2.修改表**1)增加新的列**

```
alter table emp1  
add(birthday date)
```

2)修改现有的列

```
alter table emp1  
modify(name varchar2(25) default 'abc')
```

3)重命名现有的列

```
alter table emp1
```

```
rename column salary to sal;
```

4) 删除现有的列

```
alter table emp1  
drop column birthday;
```

3. 清空表中的数据(与 delete from table_name 区分开)

```
truncate table emp2;
```

4. 重命名表

```
rename emp2 to emp3;
```

5. 删除表

```
drop table emp3;
```

数据处理 DML

1) 增

1.1 增添一条记录

```
insert into [表名](,,,,)  
values(,,,,)
```

1.2 从其它表中拷贝数据

```
insert into [表名]  
select .... from [另一个表]  
where ....
```

2) 改

```
update [表名]  
set ....  
where ....
```

3) 删

```
delete from [表名]  
where ....
```

4) 查(最常用的数据库操作)

```
select ....  
from ...  
where ....  
group by ...  
having ...  
order by ....
```

约束

对创建的表的列属性、字段进行的限制。诸如: not null/unique/primary key/foreign key/check

1. 如何定义约束--在创建表的同时，添加对应属性的约束

1.1 表级约束 & 列级约束

```
create table emp1(  
    employee_id number(8),  
    salary number(8),  
    --列级约束  
    hire_date date not null,  
    dept_id number(8),  
    email varchar2(8) constraint emp1_email_uk unique,  
    name varchar2(8) constaint emp1_name_uu not null,  
    first_name varchar2(8),  
    --表级约束  
    constraint emp1_emp_id_pk primary key(employee_id),  
    constraint emp1_fir_name_uk unique(first_name),  
    constraint emp1_dept_id_fk foreign key(dept_id) references  
departments(department_id) ON DELETE CASCADE  
)
```

1.2 只有 **not null** 只能使用列级约束。其他的约束两种方式皆可

2. 添加和删除表的约束--在创建表以后，只能添加和删除，不能修改

2.1 添加

```
alter table emp1  
add constaint emp1_sal_ck check(salary > 0)
```

2.1.1 对于 **not null** 来讲，不用 **add**，需要使用 **modify**:

```
alter table emp1  
modify (salary not null)
```

2.2 删除

```
alter table emp1  
drop constaint emp1_sal_ck
```

2.3 使某一个约束失效:此约束还存在于表中，只是不起作用

```
alter table emp1
```

```
disable constraint emp1_email_uk;
```

2.4 使某一个约束激活：激活以后，此约束具有约束力

```
alter table emp1
```

```
enable constraint emp1_email_uk;
```

尚硅谷数据库视频教程下载，请访问 <http://www.atguigu.com/download.shtml#1>