Oracle 数据库笔记

笔记中用到的表>>

SALGRADE(薪资)表：

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

EMP(员工)表：

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	ROWID
1	2222	张翼麒							AAAR3sAAEAAAACWAAA
2	3333	张翼麒							AAAR3sAAEAAAACWAAB
3	7369	SMITH	CLERK	7902	1980-12-17	1800.00		20	AAAR3sAAEAAAACXAAA
4	7499	ALLEN	SALESMAN	7698	1981-2-20	1600.00	300.00	30	AAAR3sAAEAAAACXAAB
5	7521	WARD	SALESMAN	7698	1981-2-22	1250.00	500.00	30	AAAR3sAAEAAAACXAAC
6	7566	JONES	MANAGER	7839	1981-4-2	3975.00		20	AAAR3sAAEAAAACXAAD
7	7654	MARTIN	SALESMAN	7698	1981-9-28	1250.00	1400.00	30	AAAR3sAAEAAAACXAAE
8	7698	BLAKE	MANAGER	7839	1981-5-1	2850.00		30	AAAR3sAAEAAAACXAAF
9	7782	CLARK	MANAGER	7839	1981-6-9	2450.00		10	AAAR3sAAEAAAACXAAG
10	7788	SCOTT	ANALYST	7566	1987-4-19	14400.00		20	AAAR3sAAEAAAACXAAH
11	7839	KING	PRESIDENT		1981-11-17	5000.00		10	AAAR3sAAEAAAACXAAI
12	7844	TURNER	SALESMAN	7698	1981-9-8	1500.00	0.00	30	AAAR3sAAEAAAACXAAJ
13	7876	ADAMS	CLERK	7788	1987-5-23	2100.00		20	AAAR3sAAEAAAACXAAK
14	7900	JAMES	CLERK	7698	1981-12-3	950.00		30	AAAR3sAAEAAAACXAAL
15	7902	FORD	ANALYST	7566	1981-12-3	4000.00		20	AAAR3sAAEAAAACXAAM
16	7934	MILLER	CLERK	7782	1982-1-23	1300.00		10	AAAR3sAAEAAAACXAAN
*									

DEPT(部门)表：

	DEPTNO	DNAME	LOC
1	10	ACCOUNTING	NEW YORK
2	20	RESEARCH	DALLAS
3	30	SALES	CHICAGO
4	40	OPERATIONS	BOSTON

A(测试索引效率的百万数据)表：

	ID	TEXT
345022	349055	5D1487CD309E4DCD859934C6E03F4277
345023	349056	B885D2F045964AAA8B6AE4ACEBCF7C30
345024	349057	D81FB0C180E54411928C723478BF7CB5
345025	349058	D37B63DEF31749C283D361A58ABFA6A2
345026	349059	585EE9CD38904021934AD799C7C317B2
345027	349060	88E56B44D8F14735866BA92FABEF7DE1
345028	349061	AC605D039D53483D8007E1C6E9290F47
345029	349062	9A404CC44ECD4DC6A44C6B20029B9BCE
345030	349063	7982D21890D848A2B4B8D84AB7541C34
345031	349064	92C283FDB67448088EE5438612379CF9
345032	349065	EA085116D7B947EA9A79EA9AEB5B1F46
345033	349066	0FD26A957929457484173D88BD2BD282
345034	349067	A09A8DD5D66F49B9AD5422723CB94D0A
345035	349068	4FCDCD2A67894556A2956F1FD62FEBA6
345036	349069	E65DAF12DF6043C491FCE8EF314EB8F1
345037	349070	77666A70360B4E15B590C11BE1B4AA18
345038	349071	F85A195632DB42E38C8B70E8DF1EB815
345039	349072	07E1ECC4B3CF443CA5E480BDE6805AF8
345040	349073	F9099B003EF24ACDB10F9999B48B06F9
345041	349074	3BE0222710664498BF4AA6F8E64E3FE9
345042	349075	5898321EBD164175A038BD3B9C629EBB
345043	349076	F2E8E74154B94B069DB01A0674A9E66D
345044	349077	6C9FC43160C14E0B96E47F3A51F098F4
345045	349078	D0D961FF5446434CB0C7D4E0F5C9E76E
345046	349079	1A60431150B34E76B179510CDFE52C80
345047	349080	A77B533A5D4D47968D56B7883EA5A3E

--oracle 表空间的创建：

```
create tablespace ZYQ
detafile 'd:/tablespaces/qq.dbf'
size 300M
autoextend on    --开启自动扩张
next 30M;
```

```
--用户的创建
create user zyq identified by 123456
default tablespace ZYQ;
--权限赋予：
grant resource to zyq;
--权限收回
revoke resource from zyq;
--权限查看
select * from session_privs;

--[创建表]
create table user_1(
   u_id number,
   u_name varchar2(10)
);
--[修改表]
--1、添加列：
alter table user_1 add u_age number(5);
--2、修改列属性:
alter table user_1 modify u_age number(10) not null;
--3、删除列：
alter table user_1 drop column u_age;
--4、修改列名
alter table user_1 rename column u_age to u_age2;

--[约束]
create table user_2(
    u_id number primary key,
    u_name varchar2(20) not null,
    u_age number default 20 check(u_age between 1 and 100),
    u_address varchar2(10) unique,
    u_sex char(2) check(u_sex in ('男','女'))
);

--DML 语句：
insert into user_1 values(1,'张翼麒',20);

insert into user_1(u_name,u_age2) values('麒麒哥',22);

select * from user_1;--【查询user_1】

update user_1 set u_name='牛逼',u_age2=30 where u_id=1;
```

```
--[序列]
--创建序列
create sequence ZZ;
--其中有两个属性 nextval currval
select ZZ.Nextval from dual;--自增
--以后属性需要自增 例如
insert into user_1(u_id,u_age2) values(zz.nextval,33);--添加 id 为原来的值+1

--【系统函数】
--1、单行函数
--字符串函数
lower:--转换为小写
upper:--转换为大写
select lower(ename) from emp;--查询 emp 表 转换为小写
select upper(ename) from emp;--查询 emp 表 转换为大写
--日期函数
select sysdate-hiredate from emp;--现在的系统时间减 emp 表入职时间 得到天数
select (sysdate-hiredate)/7 from emp;--周数
select months_between(sysdate,hiredate) from emp;
--数值函数(四舍五入)
select round(2.69) from dual;
select round(2.69,1) from dual;
select round(2.69,2) from dual;
--通用函数 求工资*12+奖金 需要加上 nvl 函数 当奖金为空，就求工资*12+0
select sal*12+nvl(comm,0),sal,comm from emp;

--转换函数
--to_char 把当前时间转化为字符串
select to_char(sysdate ,'yyyy-MM-dd hh24:mi;ss day')from dual;
--to_date 把字符串转化为时间
select to_date('2019-06-26 21:26;56 星期三','yyyy-MM-dd hh24:mi;ss day')from dual;
--eg
insert into emp(empno,hiredate) values(2,to_date('2017-07-11','yyyy-MM-dd'));
select *from emp;


--【分组查询】
--先查询每个部门的人数
select deptno,count(*) from emp group by deptno;
--大于 5 人的部门
select deptno,count(*) from emp group by deptno having count(*)>5;
--部门的信息需要关联另外一张表
select e.deptno,d.dname,d.loc,count(*) from emp e,dept d where e.deptno=d.deptno group by
e.deptno,d.dname,d.loc having count(*)>5;
```

```
--【多表查询】
--内连接
--查询全部员工的领导信息
select * from emp;--[emp 表为例] 7369 员工的领导是 7902  存在于同一张表
select * from emp e,emp m where e.mgr=m.empno ;--同一张表起别名
--在上面基础上查询出员工部门信息 ， 需要引入部门表 dept
select * from emp e,dept d where e.deptno=d.deptno;
--查询领导信息及其部门信息
select * from emp e,emp m,dept d1 where e.mgr=m.empno and m.deptno=d1.deptno;
--如果需要查询员工和领导的部门 则需要两张 dept 表 防止只查询员工和领导相同的部门
select * from emp e,emp m,dept d1 ,dept d2 where e.mgr=m.empno and e.deptno=d1.deptno and
m.deptno=d2.deptno;
--查询员工领导的工资等级 需要用工资去工资表比对
select * from emp e,salgrade s,emp m where e.mgr=m.empno and m.sal between s.losal and
s.hisal;


--外连接
--查询员工领导
select * from emp e left join emp m on e.mgr=m.empno;--左外连接
select * from emp m right join emp e on e.mgr=m.empno; --右外连接
select * from emp e left join dept d on e.deptno=d.deptno;
select * from emp e left join emp m on e.mgr=m.empno left js，oin dept d on m.deptno=d.deptno;


--【子查询-内查询的结果是外查询的条件】
--一个查询语句包含另一个查询语句
select * from emp;
--例子 1:查询与 7499 员工工作一致的员工【返回一个值 适用于= > <这些符号处理】
--1、先查询 7499 的工作
select job from emp where empno=7499;
--2、查询与 7499 员工工作相同的员工【返回一列 适用于 in】
select * from emp where job='SALESMAN'--这样可以
select * from emp where job=(select job from emp where empno=7499);--整合(子查询)

--例子 2：查询存在员工的部门信息 思路：只要员工表里存在 deptno 则说明 该部门有人
select deptno from emp;--查询出全部
select distinct deptno from emp;--去重
select * from dept where deptno in(10,20,30);--查询 10 20 30  这三个部门
select * from dept where deptno in(select distinct deptno from emp);

--例子 3：查询员工薪水大于本部门的平均薪水的员工的信息【如果返回的是一张表 作为
```

```
表来使用】
--思路 查询每个部门的平均薪水  再查询那些员工的薪水大于平均薪水 必须是同一个部门
select avg(sal),deptno from emp group by deptno;--查询平均工资
select * from emp e,(select avg(sal) a,deptno from emp group by deptno) t where e.sal>t.a and e.deptno=t.deptno;


--【分页查询】
--oracle 没有 limit 关键字
--伪列：虚拟的列
--rownum 行号 在生成行数据时候生成，从 1 开始 没有上限
--例子 1：生成行号
select e.* ,rownum from emp e;
--查询前三条记录
select e.* ,rownum from emp e where rownum<=3;
--查询 4-6 条记录
select e.* ,rownum from emp e where rownum between 4 and 6;--错误的 原理：行号是生产数据时生成，一直生成的都是行号 1 所以查不出数据
--解决
select t.* from (select e.* ,rownum r from emp e) t where t.r between 4 and 6;

--例子 2：工资前三的信息
--思路：工资降序排序 取前三
select   * from emp order by sal desc;--工资降序
select   e.* ,rownum from emp e order by sal desc;--出现问题 先生成行号后排序 行号乱了
--解决：先排序 作为表再生成行号
select e.* ,rownum from (select   * from emp order by sal desc) e;
--最后加上条件取前三
select e.*,rownum   from(select * from emp order by sal desc) e where rownum <=3;

--例子 4：工资 4-6 名
--思路和例子 3 相似，把例子三再作为一张表
select t.* from（select e.*,rownum r   from(select * from emp order by sal desc) e） t where t.r between 4 and 6;
/*
一、【视图】
   什么是视图：能看到的 可视化 是一个虚表 封装了复杂的 sql 查询语句
   可以对视图进行增删改查 不能存储数据 数据都存储再基本表中

   视图的语法：
   create or replace view 视图名 as 比较复杂的 sql 语句

*/
```

```
create or replace view emp_view as select * from emp;
--进行增删改查
select * from emp_view;
insert into emp_view(empno ,ename) values (2222,'张翼麒');
--操作视图就是操作基本表
--作用1、屏蔽敏感列（密码 工资等）
create or replace view view1 as select empno,ename, job ,mgr ,hiredate, deptno from emp;
select * from view1;--这样查出的视图就只有想让人看到的
--可以弄成只读的视图就不能执行增删改查啦
create or replace view emp_view2 as select *from emp with read only;
insert into emp_view2(empno ,ename) values (2222,'张翼麒');--无法对只读的视图经行增删改



/*
二、【索引】
 目录
 创建索引的三前提：百万条记录以上；经常查询使用的列；不经常修改的列：经常修改的列会导致索引重建
 所用：提高查询的xiaol
 语法：create index 索引名称 on 表(列.....)
 注意：主键列 和 唯一列 本身就是索引列 不能添加索引
*/
--测试单例索引的效率
--1、创建表
create table A(
    id number primary key,
    text varchar2(200)
)
--2、添加百万条数据
create sequence asue;--创建序列 自增使用
select sys_guid() from dual;--一个无序的串，存在 text 中

--使用for 循环添加百万次
declare

begin
   for i in 1..1000000 loop
   insert into A values(asue.nextval,sys_guid());
   end loop ;

end;
--3、添加索引前查询
select * from A where text='B52FB10823714208A184C427D147797C';--0.040s
```

```
--4、添加索引
create index a_index on A(text);
--5、索引后查询
select * from A where text='18C291A532C343C6BAAB36BEE1B1F158';

--测试复合索引效率（给两列以上创建索引)
--eg：表名(name ,addr)
select * from 表 where name='' and addr='';--可以触发索引
select * from 表 where name='' or addr='';--不可以触发索引 or 会进行全局扫描，索引将会无意义
select * from 表 where name='';--可以触发索引
select * from 表 where addr='';--不可以触发索引 有先后顺序



/*
三、【PLSQL 过程化语言】
p:procedure L:language
在 sql 语言的基础上新增加了 if 循环

1、PLSQL 的基本结构：
declare -- 声明
   声明变量的内容
begin -- 开始
   过程化语言 sql
end； -- 结束
*/

--[声明变量]
declare
--声明普通变量
i number; -- 下面初始化
j number default 200;--直接初始化
l emp.ename%type;--若不知道查出来将要赋值的数据的类型 可以直接写 [表.列%type] 例如：k emp.ename%type;
k emp.job%type;
e emp%rowtype;--声明一行的类型
begin
   i:=100;--i 需要经行初始化
   dbms_output.put_line(i);
   dbms_output.put_line('j='||j);--连接符

   --select ename from emp where empno=7788; 想把查询出来的数据传入控制输出
   -- 可以使用 select ...into...:把查到的数据赋值给某些变量
```

```
    select ename into l from emp where empno=7788;
    dbms_output.put_line(l);

    --查询两列 赋值两个变量
    select ename,job into l , k from emp where empno=7788;
    dbms_output.put_line(l||'-'||k);

    --查询一行的数据
    select * into e from emp where empno=7788;--能接收全部的数据
      dbms_output.put_line(e.empno||'-'||e.ename);
end;

--[if]
--eg:输入数字>0 ...x<0...
declare
i number;
begin
    i:=&请输入;
    if i>0 then
        dbms_output.put_line('正数');
        elsif i<0 then
            dbms_output.put_line('负数');
        else
             dbms_output.put_line('0');
             end if;--Oracle 都会用的结束
end;
--[for]
--eg:
declare
begin
    for i in 1..10 loop
          dbms_output.put_line(i);
    end loop;
end;
--[while]
declare
i number default 1;
begin
    while i<=10 loop
        dbms_output.put_line(i);
        i:=i+1;
        end loop;
    end;
```

```
/*
四、【游标】
 游标就是一个集合 存储大量数据
 什么时候使用游标  select into 只能处理一行数据 赋值给变量
                   如果要处理多行数据使用游标

 声明游标：cursor 游标名 is sql 查询语句
 遍历游标：
 a、打开游标：open 游标名
 b、提取一行数据：fetch 游标名  into 变量 ；把游标中一行的数据读入变量中
 c、循环，遍历所有数据，退出：exit when 游标名%notfound：(循环至再也提取不到数据 就停止)
 d、关闭游标 close 游标名
*/

--例子 1：在控制输出 20 号部门的全部员工信息-使用游标（使用 loop 循环）
--分析：把所有员工存入游标中遍历游标
declare
  --声明游标
  cursor cur is
          select * from emp where deptno=20;
  --声明变量存放一行的数据
  e emp%rowtype;
begin
  --遍历游标
  --a、打开游标
  open cur;
  loop
  --b、提取一行数据：fetch 游标名  into 变量 ；把游标中一行的数据读入变量中
  fetch cur into e;
  --c、循环，遍历所有数据，退出：exit when 游标名%notfound：(循环至再也提取不到数据 就停止)
    exit when cur%notfound;
  dbms_output.put_line('编号-'||e.empno||'姓名'||e.ename);
  end loop;
  -- d、关闭游标 close 游标名
  close cur;
end;

--例子 2：给 20 号部门的员工涨工资
--使用 for 循环 会自动开启关闭游标
declare
```

```
    cursor cur is
            select empno from emp where deptno=20; --查询全部员工的编号 下面循环去除
编号 根据编号涨工资
begin
  for i in cur loop   --i 是指一行的数据
    --dbms_output.put_line('编号-'||i.empno); 查询看看编号
    update emp set sal=sal+1000 where empno=i.empno;
    end loop;
  end;

select * from emp where deptno=20;



/*
五、【存储过程】
理解：把过程存储起来
封装了一组 sql 语句 提前编译好 存放在服务端
安全 效率高

存储过程使用场景：买了一件商品>数据库的变化
订单表（insert），商品表(库存-1)，购物车(清空 delete)，余额(减少--)，物流表(insert),日志
表(insert)

存储过程的语法：参数分类 输入型参数和输出型参数
create or replace procedure 过程名称（参数列表：参数 1 in||out 数据类型，参数 2 in||out 数
据类型....）
is||as    相当于 plsql 中的 declare：声明变量
begin
  end;

*/
--例子 1：给某员工涨工资
--分析：需要>员工编号 输入型 number
--              涨多少   输入型 number
create or replace procedure updateSal(eno in number,psal in number)
as
oldsal emp.sal%type;
newsal emp.sal%type;
begin
    select sal into oldsal from emp where empno=eno; --查询涨工资之前并在控制台打印
    dbms_output.put_line(oldsal);
  update emp set sal=sal+psal where empno=eno;--涨工资
    select sal into newsal from emp where empno=eno; --查询涨工资之后并在控制台打印
  dbms_output.put_line(newsal);
```

```
end;
--执行存储过程涨工资
call updateSal(7788,10000);


--例子2：查询某员工的年薪
--分析：需要>员工的编号 输入型 number
--              年薪 输出型 number
create or replace procedure FindYearSal(eno in number,yearsal out number)
as
begin
   select sal*12+nvl(comm,0) into yearsal from emp where empno=eno;
   end;

--执行存储过程   有输出的存储过程   需要配合PLsql
declare
--需要声明一个变量接收年薪
 Ysal number;
begin
   FindYearSal(7788,Ysal);
   dbms_output.put_line(Ysal);
   end;

--例子3：输出类型为游标类型(查询某部门的所有员工的信息)
--分析：    需要>部门编号 in number
--                  全部员工信息 out 游标类型
--游标类型：静态游标类型(cursor) 声明游标时候指定sql语句
--              动态游标类型(sys_refcursor) 声明式不指定sql语句 在使用时候指定
create or replace procedure AllEmps(eno in number ,emps out sys_refcursor)
as
begin
    open emps for select * from emp where deptno=eno;--打开游标 将全部员工存入
   end;
--访问储存过程
declare
 emps sys_refcursor;
 e_row emp%rowtype;--声明变量接收游标内容
begin
    AllEmps(20,emps);
    --遍历游标
    loop--循环
    fetch emps into e_row;
    --退出循环的条件
    exit when emps%notfound;
```

```
    dbms_output.put_line('编号:'||e_row.empno||'-姓名:'||e_row.ename);
    end loop;
  end;

/*
六、【触发器】
  a、什么式触发器
  监听器：监听表中数据是否有改变；
  监听：增、删、改
  语法：
     create or replace trigger  触发器名称
     before|after   触发器在改变数据之前执行还是之后执行
     insert | update |delete     监听的动作是什么
     on   表          监听什么表
     触发器级别
     declare

     begin

       end;
*/
--例子1：添加一条记录 在控制天打印 '添加了一条数据'
create or replace trigger AddOne
after
insert
on emp
declare
begin
   dbms_output.put_line('添加了一条记录');
   end;
--添加
insert into emp(empno,ename) values (3333,'张翼麒');

--例子2：不能给员工降薪
create or replace trigger NotLowSal
before
update
on emp
--触发器级别
    --a、表级触发器(默认)：更新多条数据执行一次触发器
    --b、行级触发器：每更新一条记录执行一次
    --注意：使用 old new 关键字必须 使用行级触发器
for each row--行级触发器
declare
```

```
begin
   --判断涨前工资和涨后工资
   if :old.sal>:new.sal    then
      --弹出错误提示  不能降薪
      raise_application_error(20001,'不能降薪~~~');
      end if;
   end;

--测试更新
update emp set sal=sal-200 where empno=7788;
```