Oracle函数

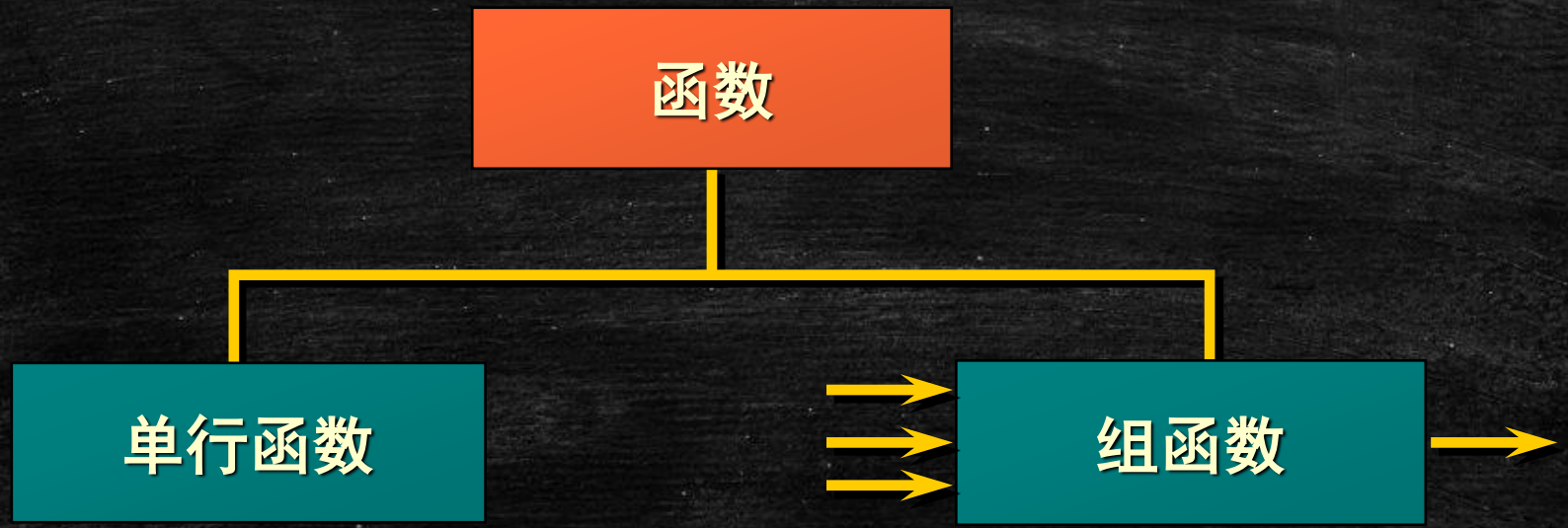
- What?Why?How?

SQL 函数

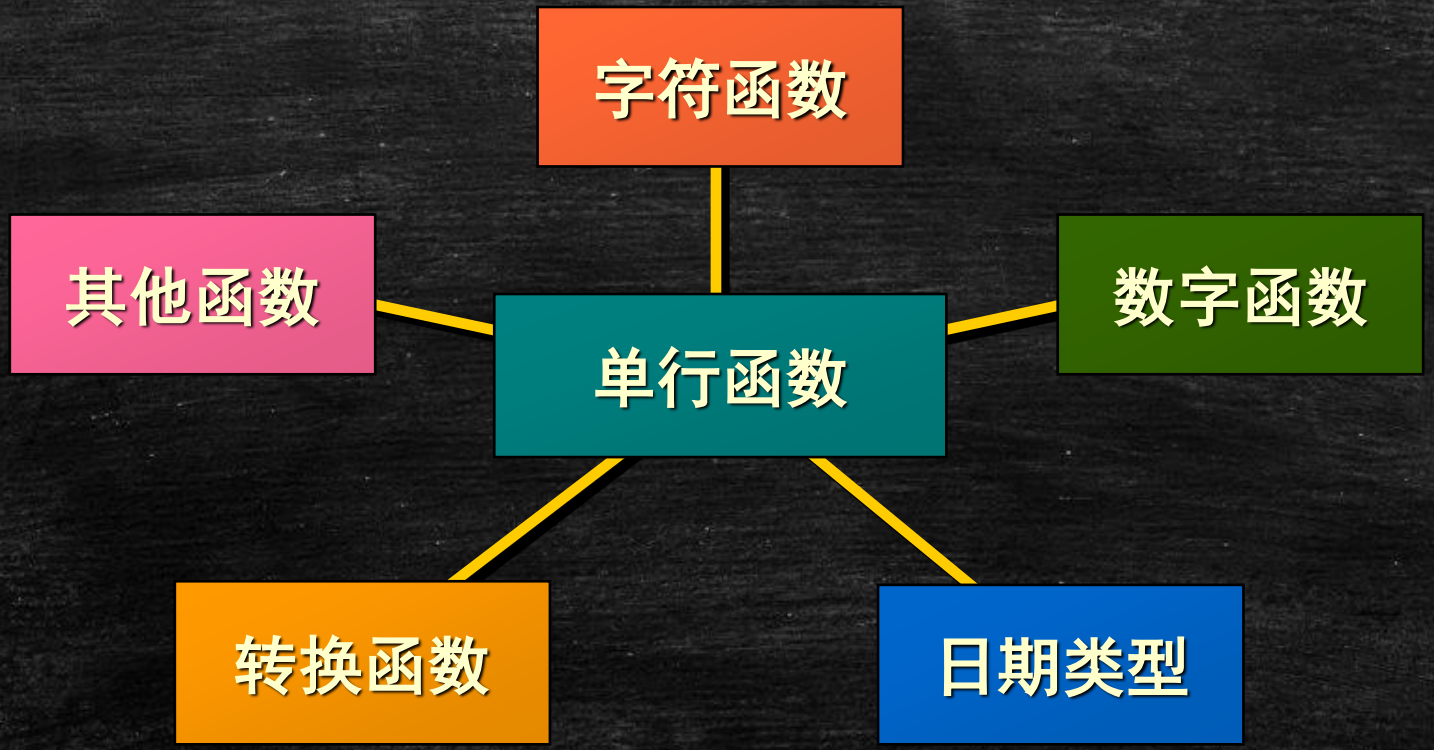
- 函数一般是在数据上执行的，它给数据的转换和处理提供了方便。只是将取出的数据进行处理，不会改变数据库中的值。

SQL 函数

- Sql函数可以分为组函数和单行函数。
  - 组函数又被称作聚合函数，用于对多行数据进行操作，并返回一个单一的结果，组函数仅可用于选择列表或查询的having子句
  - 单行函数对单个数值进行操作，并返回一个值。



单行函数的分类



字符函数

- 字符函数全以字符作为参数，返回值分为两类：一类返回字符值，一类返回数字值
  - concat（string1, string2）连接两个字符串 ||
  - initcap（string）string中每个单词首字母大写
  - Lower(string) 以小写形式返回string
  - lpad,rpad 填充字符型数据
  - ltrim/rtrim （string1,string2)
  - trim(A from B)
  - Substr（） 提取字符串的一部分substr（string, 1, 2)
  - upper(string) 以大写形式返回string
  - Instr（）字符串出现的位置，instr（ string , 'A ‘）
  - Length （）字符串长度

字符函数

```
select upper(lower(ename)) from emp;
select substrb('中aa国人abc', 1,4) from dual;
select ename,substr(ename, 2) from emp;
select ename, length(ename) from emp;
select ename, instr(ename, 'A') from emp;
select ename, lpad(ename,'6','**') from emp;
select ename,replace(ename,'A','***') from emp
select RTRIM('gao qian jingXXXX','X') text from dual;
select trim(both 'x' from 'xxxsdfsdfx') from dual;
```

数字函数

- 数字函数以NUMBER类型为参数返回NUMBER值
- round（number，n）返回四舍五入后的值
  - select round(23.652) from dual;
  - select round(23.652, 2) from dual;
  - select round(23.652, -1) from dual;
- trunc（number，n）
  - select trunc(23.652) from dual;
  - select trunc(23.652, 2) from dual;
  - select trunc(23.652, -1) from dual;
- mod（x，y）求余数
  - select mod(13,5) from dual;
- ceil()上取整 select ceil(19.2) from dual;
- floor()下取整 select floor(19.2) from dual;

时间函数

- select current_time() from dual;---- mysql:时间。
- select current_date() form dual; ---mysql；日期
- select current_timestamp() from dual;---mysql：日期时间

日期和时间函数

- Oracle以内部数字格式存储日期:世纪,年,月,日,小时,分钟,秒
  - sysdate/current_date

    以date类型返回当前的日期
  - Add_months(d,x) 返回加上x月后的日期d的值
  - LAST_DAY(d) 返回的所在月份的最后一天
  - Months_between(date1,date2) 返回date1和date2之间月的数目
    - 工作年限30以上

日期和日期时间算术

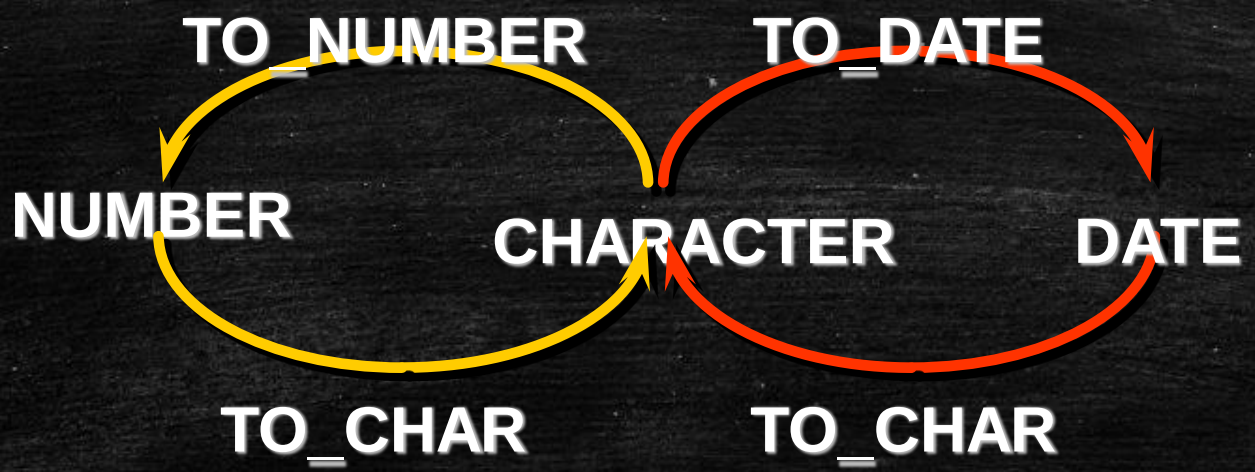
- 从日期中加或减一个数值, 以得当一个日期结果值
  - select sysdate+2 from dual;
  - select sysdate-2 from dual;
- 两个日期相减以便得到他们相差多少天
  - select ename,round((sysdate-hiredate)/7) weeks from emp where deptno=10

转换函数

- 标量数据可以有类型的转换，转换分为两种，隐式类型转换和显示类型转换。
- 隐式类型转换可用于：
  - 字符和数字的相互转换 &字符和日期的相互转换
  - VARCHAR2 or char - - number
  - VARCHAR2 or char - - date
  - number - - varchar2
  - date - - varchar2
  - select * from emp where empno=to_number('8000')
  - select * from emp where hiredate='20-2月-1981'
- 尽管数据类型之间可以进行隐式转换，仍建议使用显示转换函数，以保持良好的设计风格。
- Select '999'-10 from dual;

转换函数

- to_char
- to_number
- to_date



TO_CHAR 函数操作日期

格式元素	含义
YYYY、YY	代表四位、两位数字的年份
MM	用数字表示的月份
MON	月份的缩写、对中文月份来说就是全称
DD	数字表示的日
DY	星期的缩写，对中文的星期来说就是全称
HH24，HH12	12小时或者24小时进制下的时间
MI	分钟数
SS	秒数

- TO_CHAR(*date*, '*fmt*')
- 用于将日期或时间戳转换成varchar2类型字符串，如果指定了格式字符串，则用它控制结果的结果。
  - 格式控制串由格式元素构成。
  - 格式控制串必须用单引号括起来

TO_CHAR 函数操作日期

- Select to_char(sysdate, 'dd-mon-yy hh24:mi:ss') "Rigth Now" from dual;
- select ename, hiredate, to_char(hiredate,'yyyy/mm/dd') from emp
- select sysdate, to_char(sysdate,'yyyy-mon-dd hh12:mi:ss') from dual;

TO_CHAR 函数操作数字（A）

控制符	含义
9	代表一位数字，如果该位没有数字则不进行显示，但对于小数点后面的部分仍会强制显示
0	代表一位数字，如果该位没有数字则强制显示0
$	显示美元符号
L	显示本地货币符号
.	显示小数点
,	显示千分位符号

- to_char（num,format）
- 用于将Number类型参数转换为varchar2类型，如果指定了format，它会控制整个转换。

TO_CHAR 函数操作数字

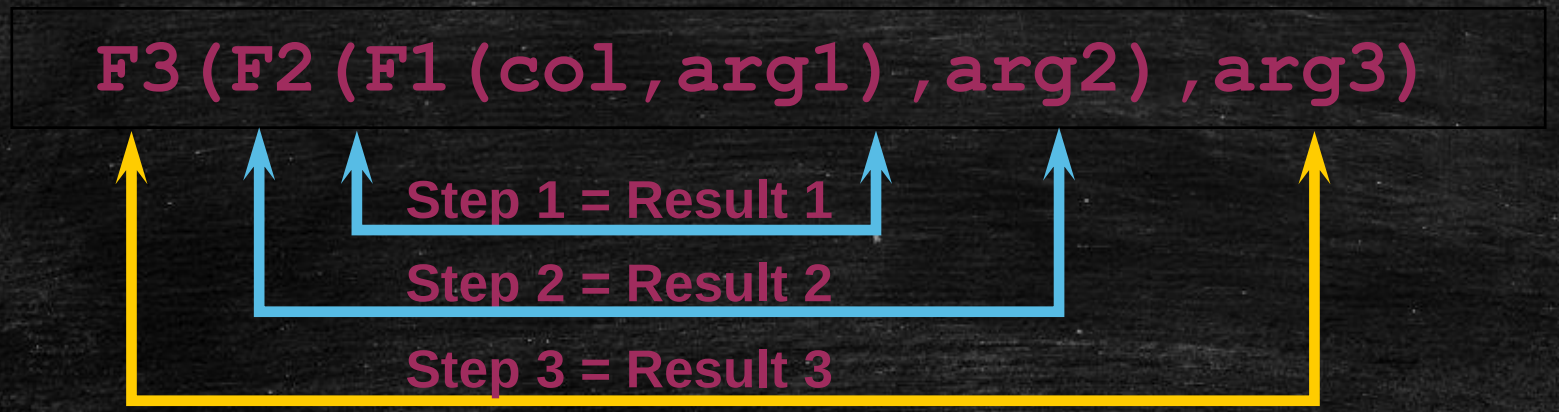
- select to_char(sal, ‘$99,999.9999’) salary from emp where ename = ‘ALLEN’;
- select to_char(sal, ‘$00,000.0000’) salary from emp where ename = ‘ALLEN’;
- select to_char(123456, '99,99,00') from dual;

to_number & to_date

- to_date (String,format)
  - 将char或varchar2类型的string转换为date类型
  - Select to_date('04,05,19,10,23,40','yy,mm,dd,hh12,mi,ss') from dual;
  - select to_date('2004-09-19','yyyy-mm-dd') from dual;
- to_number(String,format)
  - 将char或varchar2类型的string转换为number类型
  - select to_number('$39343.783','$99990.000') from dual;
  - select to_number('11.231','999.999') from dual;

单行函数嵌套

- 单行函数可被嵌入到任何层
- 嵌套函数从最深层到最低层求值



单行函数嵌套

- 显示没有上级管理的公司首脑
  - 没有上级领导的雇员 mgr显示为boss
  - select ename,nvl(to_char(mgr),’no manager’)
    from emp
    where mgr is null;
- 显示员工雇佣期满6个月后下一个星期五的日期
  - Select to_char(next_day(add_months(hiredate,6),’Friday’),’ fmDay,Month ddth,YYYY’) “review”
    from emp order by hiredate;

其他函数

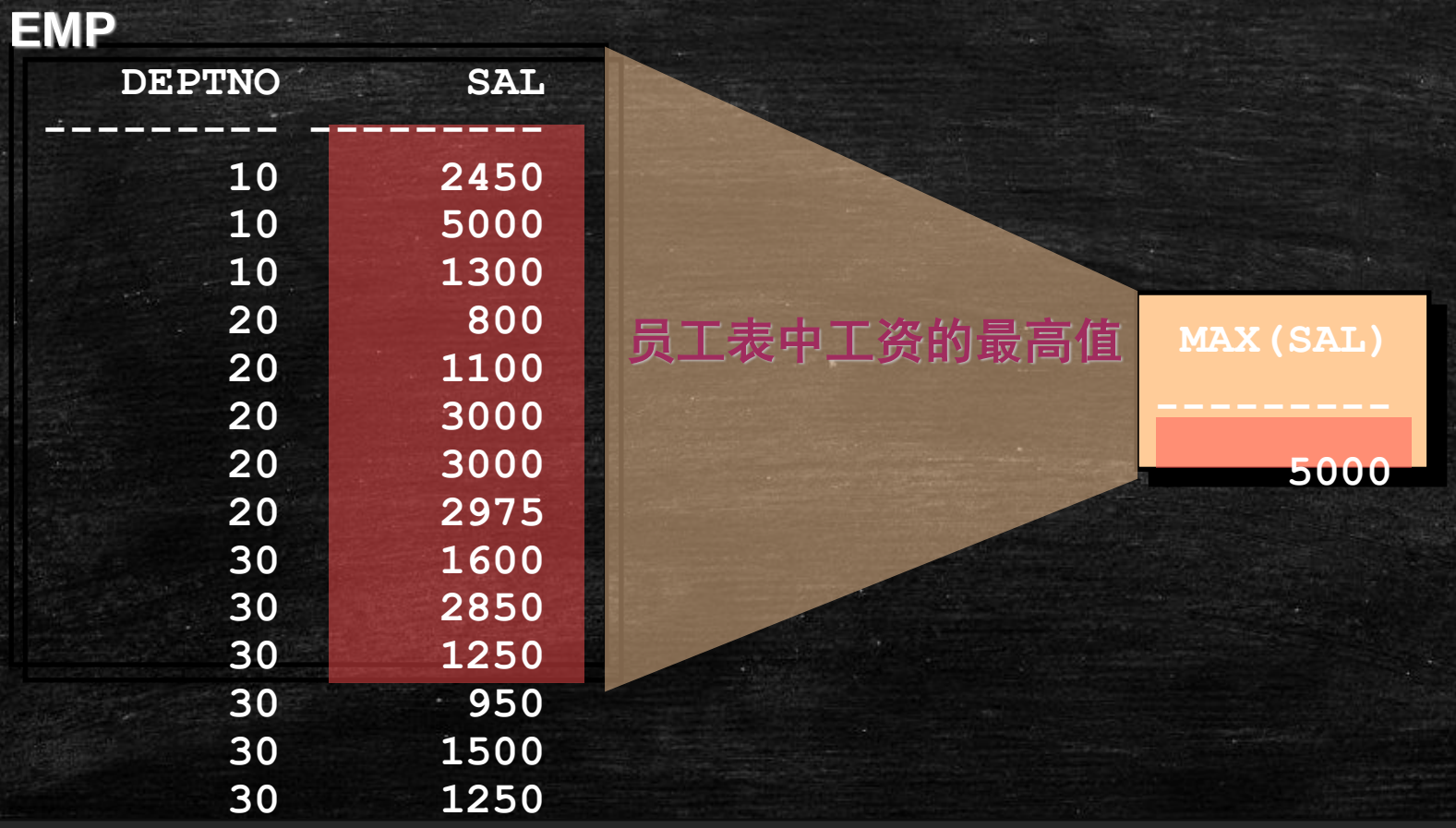
decode
case when

课堂练习

- 1、查询82年员工
- 2、查询37年工龄的人员
- 3、显示员工雇佣期 6 个月后下一个星期一的日期
- 4、找没有上级的员工，把mgr的字段信息输出为 "boss"
- 5、为所有人长工资，标准是：10部门长10%；20部门长15%；30部门长20%其他部门长18%

组函数（A）

- 组函数基于多行数据返回单个值



组函数（A）

- avg()→返回某列的平均值
- min()→返回某列的最小值
- max()→返回某列的最大值
- sum()→返回某列值的和
- count()→返回某列的行数
- **组函数仅在选择列表和Having子句中有效**

组函数（A）

- 在数字类型数据使用AVG and SUM 函数
  - select sum(sal), avg(sal), max(sal) , min(sal) from emp;
- MIN and MAX适用于任何数据类型
  - select min(hiredate) ,max(hiredate) from emp;
- 组函数除了count(*)外，都跳过空值而处理非空值
  - select count(*) from emp;
  - select count(comm) from emp;
  - select count(1) from emp;
  - 不能计算空值
- select count(distinct deptno) from emp;

在分组函数中使用NVL函数

- 组函数不能处理null
- select avg(comm) from emp;
- NVL函数迫使分组函数包括空值
- select avg(nvl(comm,0)) from emp;

数据分组

- 创建分组
  - group by 子句
  - Group by 子句可以包含任意数目的列。
    - 除组函数语句外，select语句中的每个列都必须在group by 子句中给出。
  - 如果分组列中具有null值，则null将作为一个分组返回。如果列中有多行null值，他们将分为一组。
  - Group by 子句必须出现在where子句之后，order by 子句之前。
- 过滤分组（having子句）
  - Where过滤行，having过滤分组。
  - Having支持所有where操作符。
- 分组和排序
  - 一般在使用group by 子句时，应该也给出order by子句。

数据分组（A）

- SELECT *column, group_function*
- FROM *table*
- [WHERE *condition*]
- [GROUP BY *group_by_expression*]
- [ORDER BY *column*];
- [having condition]
- 使用GROUP BY子句将表分成小组
- 结果集隐式按降序排列,如果需要改变排序方式可以使用Order by 子句

数据分组

- 出现在SELECT列表中的字段，如果出现的位置不是在组函数中，那么必须出现在GROUP BY子句中
- select deptno,avg(sal) from emp group by deptno
- GROUP BY 列可以不在SELECT列表中
- select avg(sal) from emp group by deptno
- 不能在 WHERE 子句中使用组函数. 不能在 WHERE 子句中限制组. 使用Having对分组进行限制
- select avg(sal) from emp group by deptno
  having avg(sal) > 1000;

Select子句顺序

子句	说明	是否必须使用
select	要返回的列或表达式	是
from	从中检索数据的表	仅在从表选择数据时使用
where	行级过滤	否
group by	分组说明	仅在按组计算聚集时使用
Having	组级过滤	否
order by	输出排序顺序	否

Select子句顺序

- Sql语句执行过程：
  1. 读取from子句中的基本表、视图的数据，[执行笛卡尔积操作]。
  2. 选取满足where子句中给出的条件表达式的元组
  3. 按group子句中指定列的值分组，同时提取满足Having子句中组条件表达式的那些组
  4. 按select子句中给出的列名或列表达式求值输出
  5. Order by子句对输出的目标表进行排序。

题目

- 求部门下雇员的工资>2000 人数

例子

- 部门薪水最高
- select max(sal) from emp group by deptno;
- select max(sal),deptno, job from emp group by deptno, job;
- select avg(sal) from emp

  where sal > 1200

  group by deptno

  having avg(sal) > 1500

  order by avg(sal);

案例

- 部门里面 工龄最小和最大的人找出来
- select mm2.deptno, e1.ename,e1.hiredate from emp e1,(
- select min(e.hiredate) mind,max(e.hiredate) maxd,e.deptno from emp e group by e.deptno
-)mm2 where e1.hiredate=mm2.mind or e1.hiredate=mm2.maxd

课堂练习

- 1、查询10号部门中编号最新入职的员工，工龄最长的员工的个人信息。
- 2、从“software”找到‘f’的位置，用‘*’左或右填充到15位，去除其中的‘a’。
- 3、查询员工的奖金，如果奖金不为NULL显示‘有奖金’，为null则显示无奖金
- 4、写一个查询显示当前日期，列标题显示为Date。再显示六个月后的日期，下一个星期 日的日期，该月最后一天的日期。
- 5、查询EMP表按管理者编号升序排列，如果管理者编号为空则把为空的在最前显示
- 6、求部门平均薪水
- 7、按部门求出工资大于1300人员的 部门编号、平均工资、最小佣金、最大佣金,并且最大佣金大于100
- 8、找出每个部门的平均、最小、最大薪水
- 9、查询出雇员名，雇员所在部门名称， 工资等级。

多表查询

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
202	Fay	20
205	Higgins	110
206	Gietz	110

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	LOCATION_ID
10	Administration	1700
20	Marketing	1800
50	Shipping	1500
60	IT	1400
80	Sales	2500
90	Executive	1700
110	Accounting	1700
190	Contracting	1700

EMPLOYEE_ID	DEPARTMENT_ID	DEPARTMENT_NAME
200	10	Administration
201	20	Marketing
202	20	Marketing
102	90	Executive
205	110	Accounting
206	110	Accounting

sql:1992语法的连接（A）

- 语法规则：

```
SELECT   table1.column, table2.column
FROM     table1, table2
WHERE    table1.column1 = table2.column2;
```

- 在 WHERE 子句中写入连接条件
- 当多个表中有重名列时，必须在列的名字前加上表名作为前缀
- 连接的类型：

1. 等值连接 -- Equijoin
2. 非等值连接 -- Non-equijoin
3. 外连接 -- Outer join
4. 自连接 -- Self join

92语法

- 数据来自于多张表 ,92表连接
- 注意: 明确引用同名的列，必须使用表名 或者别名区分
- 一、迪卡尔积
- select 字段列表 from 表1,表2,表3....
- 二、等值连接: 取关系列相同的记录
- select 字段列表 from 表1,表2,表3....
- where 表1.列=表2.列 and 表1.列=表3.列
- 三、非等值连接：取关系列不同的记录 != > < >= <= between and
- select 字段列表 from 表1,表2,表3....
- where 表1.列!=表2.列 and 表1.列!=表3.列
- 四、自连接:(特殊的等值连接) 列来自于同一张表,不同角度看待表
- select 字段列表 from 表1 e,表1 m
- where e.列1=m.列2
- 五、外连接: 在等值基础上，确保 一张表(主表)的记录都存在 从表满足则匹配，不满足补充null
- 1、左外: 主表在左边
- 2、右外: 主表在右边

等值连接

- 语法规则：
- SELECT table1.column, table2.column
- FROMtable1, table2
- WHERE table1.column1 = table2.column2;
- 笛卡尔积：表*表
- 主外键
- 在外键表中的映射字段称为 外键 Foreign key
- 在主键表中的唯一字段称为主键 Primary key

非等值连接

- 非等值连接
- <,>,<=,>=,!=连接时称非等值连接
- select * from emp,salgrade where sal between losal and hisal

外连接运算符是 (+)

- 外连接运算符是 (+)

笛卡尔积

- select count(*) from emp
- select count(*) from dept
- select emp.empno,dept.loc from emp,dept
- 检索出的行的数目将是第一个表中的行数乘以第二个表中的行数
- 检索出的列的数目将是第一个表中的列数加上第二个表中的列数
- 应该保证所有联结都有where子句，不然数据库返回比想要的数据多得多的数据

等值连接

EMPLOYEES

EMPLOYEE_ID	DEPARTMENT_ID
200	10
201	20
202	20
124	50
141	50
142	50
143	50
144	50
103	60
104	60
107	60
149	80
174	80
176	80

...

Foreign key
外键

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
20	Marketing
50	Shipping
50	Shipping
50	Shipping
50	Shipping
50	Shipping
60	IT
60	IT
60	IT
80	Sales
80	Sales
80	Sales

...

Primary key
主键

等值连接

- 使用 AND 操作符增加查询条件

EMPLOYEES

LAST_NAME	DEPARTMENT_ID
Whalen	10
Hartstein	20
Fay	20
Mourgos	50
Rajs	50
Davies	50
Matos	50
Vargas	50
Hunold	60
Ernst	60

...

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME
10	Administration
20	Marketing
20	Marketing
50	Shipping
50	Shipping
50	Shipping
50	Shipping
50	Shipping
60	IT
60	IT

...

等值连接

- select emp.empno,emp.ename,dept.deptno,dept.loc
  from emp,dept
  where emp.deptno=dept.deptno
  and emp.deptno=10
- select emp.empno,emp.ename,dept.deptno,dept.loc
  from emp,dept
  where emp.deptno=dept.deptno
  and ename='JAMES'

连接中使用表的别名

- 使用表的别名简化了查询
- select e.empno,e.ename,e.deptno,d.deptno,d.loc

  from emp e,dept d

  where e.deptno=d.deptno

多于两个表的连接

- 为了连接n个表，至少需要n-1个连接条件。

EMPLOYEES

LAST_NAME	DEPARTMENT_ID
King	90
Kochhar	90
De Haan	90
Hunold	60
Ernst	60
Lorentz	60
Mourgos	50
Rajs	50
Davies	50
Matos	50
Vargas	50
Zlotkey	80
Abel	80
Taylor	80

20 rows selected.

DEPARTMENTS

DEPARTMENT_ID	LOCATION_ID
10	1700
20	1800
50	1500
60	1400
80	2500
90	1700
110	1700
190	1700

8 rows selected.

LOCATIONS

LOCATION_ID	CITY
1400	Southlake
1500	South San Francisco
1700	Seattle
1800	Toronto
2500	Oxford

多于两个表的连接

- create table manager
  as
  select * from emp;

Manager ,emp ,dept

- select e.empno,e.ename,m.ename,d.loc
  from emp e,manager m,dept d
  where e.mgr=m.empno
  and e.deptno=d.deptno
  and e.job='ANALYST'

非等值连接

- select *
  from emp,salgrade
  where sal between losal and hisal

外连接

DEPTNO	DNAME	转到 "http
50	GAME	BEIJING
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

DEPTNO	ENAME
20	SMITH
30	ALLEN
30	WARD
20	JONES
30	MARTIN
30	BLAKE
10	CLARK
20	SCOTT
10	KING
30	TURNER
20	ADAMS
30	JAMES
20	FORD
10	MILLER

已选择14行。

没有雇员属于40，50部门

Dept.deptno = emp.deptno

外连接

- 为了在操作时能保持这些将被舍弃的元组，提出了外连接的概念，使用外连接可以看到不满足连接条件的记录
  - 外连接运算符是（+）
  - 有左外连接和右外连接
- 左外连接显示左边表的全部行
  - SELECT *table.column, table.column*
    FROM *table1, table2*
    WHERE *table1.column = table2.column(+);*
- 右外连接显示右边表的全部行
  - SELECT *table.column, table.column*
    FROM *table1, table2*
    WHERE *table1.column(+) = table2.column;*

外连接

- select e.ename,d.deptno,d.dname
  from emp e,dept d
  where d.deptno=e.deptno(+);
- select e.ename,d.deptno,d.dname
  from emp e,dept d
  where e.deptno(+)=d.deptno;

自连接

- 查找每个员工的上级主管
- select worker.ename||' works for '||manager.ename
  from emp worker,emp manager
  where worker.mgr=manager.empno

sql:1999语法的连接

- sql1992的语法规则暴露了这样的缺点：语句过滤条件和表连接的条件都放到了where子句中 。当条件过多时，联结条件多，过滤条件多时，就容易造成混淆。
- SQL1999修正了整个缺点，把联结条件，过滤条件分开来，包括以下新的TABLE JOIN的句法结构:
  - CROSS JOIN
  - NATURAL JOIN
  - USING子句
  - ON子句
  - LEFT OUTER JOIN
  - RIGHT OUTER JOIN
  - FULL OUTER JOIN
  - Inner outer join

交叉连接

- CROSS JOIN产生了一个笛卡尔积，就象是在连接两个表格时忘记加入一个WHERE子句一样

```
select emp.empno, emp.ename, emp.deptno, dept.loc
from emp , dept;
```

- 可以使用CROSS JOIN 来达到相同的结果

```
select emp.empno, emp.ename, emp.deptno, dept.loc
from emp cross join dept;
```

自然连接

- NATURAL JOIN子句基于两个表中列名完全相同的列产生连接
  - 两个表有相同名字的列
  - 数据类型相同
  - 从两个表中选出连接列的值相等的所有行
- select *

  from emp natural join dept

  Where deptno = 10;

  自然连接的结果不保留重复的属性

using创建连接

- select e.ename,e.ename,e.sal,deptno,d.loc
  from emp e join dept d using(deptno)
  where deptno=20
- using子句引用的列在sql任何地方不能使用表名或者别名做前缀，同样适合natural子句

使用on创建连接

- 自然连接的条件是基于表中所有同名列的等值连接
- 为了设置任意的连接条件或者指定连接的列，需要使用ON子句
- 连接条件与其它的查询条件分开书写
- 使用ON 子句使查询语句更容易理解
- select ename,dname

  from emp join dept on emp.deptno=dept.deptno

  where emp.deptno=30;

使用on创建连接三表连接

- **检索雇员名字、所在单位、薪水等级：这三个信息在三个表里面，所以只能用多表联结**
- select ename,dname,grade
  from emp
  join dept on emp.deptno=dept.deptno
  join salgrade on emp.sal between salgrade.losal and salgrade.hisal;

左外连接

- 在LEFT OUTER JOIN中，会返回所有左边表中的行，即使在右边的表中没有可对应的列值。
- select e.ename,d.deptno,d.dname
  from dept d
  left outer join emp e
  on e.deptno=d.deptno
- select e.ename,d.deptno,d.dname
  from emp e,dept d
  where d.deptno=e.deptno(+);

右外连接

- RIGHT OUTER JOIN中会返回所有右边表中的行，即使在左边的表中没有可对应的列值。
- select e.ename,d.deptno,d.dname
  from emp e
  right outer join dept d
  on e.deptno=d.deptno
- select e.ename,d.deptno,d.dname
  from emp e,dept d
  where e.deptno(+)=d.deptno;

- --inner join 默认内连接
- --on 连接表的条件
- select * from emp e inner join dept d on e.deptno=d.deptno
- select * from emp e   join dept d on e.deptno=d.deptno
- select * from emp e   join dept d using(deptno)

子查询（A）

- SQL允许多层嵌套。子查询，即嵌套在其他查询中的查询。
- SELECT *select_list*
  FROM *table*
  WHERE *expr operator*
  (SELECT *select_list*
  FROM *table*);
- 理解子查询的关键在于把子查询当作一张表来看待。外层的语句可以把内嵌的子查询返回的结果当成一张表使用。
  - 子查询要用括号括起来
  - 将子查询放在比较运算符的右边（增强可读性）

子查询的种类（A）

- 按照子查询返回的记录数，子查询可以分为单行子查询和多行子查询

- 单行子查询

- 多行子查询

单行子查询

- 子查询返回一行记录
- 使用单行记录比较运算符

Operator	Meaning
=	Equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
<>	Not equal to

单行子查询

- 我们要查询有哪些人的薪水是在整个雇员的平均薪水之上的：
  1. 首先求所有雇员的平均薪水

     ```
     select avg(sal+nvl(comm,0)) from emp
     ```

  2. 然后求：

     ```
     select ename,empno, sal, sal+nvl(comm,0)
     from emp
     where sal+nvl(comm,0)>(select avg(sal+nvl(comm,0)) from emp);
     ```

- 此处嵌套的子查询在外层查询处理之前执行

多行子查询

- 子查询返回多行行记录
- 使用集合比较运算符

运算符	含义
IN	等于列表中的任何值
some	将值与子查询返回的任意一个值进行比较
ALL	比较子查询返回的每一个值

在多行子查询中使用in

- 我们要查在雇员中有哪些人是经理人，也就是说，有哪些人的empno号在mgr这个字段中出现过，这个时候，应当首先查询mgr中有哪些号码，然后再看看有哪些人的雇员号码在此出现：

```
select empno, ename
from emp
where empno in (
    select distinct mgr from emp
);
```

在多行子查询中使用some all

- 找出部门编号为20的所有员工中收入最高的职员

```
select * from emp
where sal >= all(
      select sal
      from emp
      where deptno = 20)
and deptno = 20
```

在From子句中使用子查询

- 我们要求每个部门平均薪水的等级，可以这样考虑，首先将每个部门的平均薪水求出来，然后把结果当成一张表，再用这张结果表和salgrade表做连接，以此求得薪水等级。

1. 先求出每个部门平均薪水的表t。
2. 将t和salgrade进行关联查询就可以了。

```
select * from
salgrade s, (select deptno,avg(sal) avg_sal
                    from emp group by deptno) t
where  t.avg_sal between s.losal and s.hisal;
```

作业

- 1、求平均薪水最高的部门的部门编号
- 2、求部门平均薪水的等级
- 3、求部门平均的薪水等级
- 4、求薪水最高的前5名雇员
- 5、求薪水最高的第6到10名雇员

分页

```
--rownum  rownum 不能直接使用>
select emp.*,rownum  from emp where rownum>=5;
select * from
(
select * from emp e order by e.sal desc) t1 where rownum<=5
select *
 from (
        select rownum rn, t2.ename, t2.sal
        from (select e.ename, e.sal from emp e order by e.sal desc) t2
       where rownum <= 10
      ) t1
 where t1.rn >= 6
```

分页

```
select *
  from (select rownum rn, t2.ename, t2.sal
          from (select e.ename, e.sal from emp e order by e.sal desc) t2
                ) t1
 where t1.rn >= 6
   and t1.rn <= 10
 ---select * from t_user limit 0,10;
limit startRow,pageSize
```

练习

- 使用99语法更改相应作业：
- --1．列出所有雇员的姓名及其上级的姓名。
- --2．列出入职日期早于其直接上级的所有雇员。
- --3．列出所有部门名称及雇员
- --4．列出所有“CLERK”（办事员）的姓名及其部门名称。
- --5．列出从事“SALES”（销售）工作的雇员的姓名，假定不知道销售部的部门编号。
- --6．列出在每个部门工作的雇员的数量以及其他信息。
- --7．列出所有雇员的雇员名称、部门名称和薪金。
- --8.求出部门编号为20的雇员名、部门名、薪水等级

日期格式的转换

- mysql日期和字符相互转换方法
- date_format(date,'%Y-%m-%d') ---------->oracle中的to_char();
- str_to_date('date','%Y-%m-%d') ---------->oracle中的to_date();
- %Y：代表4位的年份
- %y：代表2为的年份
- %m：代表月, 格式为(01……12)
- %c：代表月, 格式为(1……12)
- %d：代表月份中的天数,格式为(00……31)
- %e：代表月份中的天数, 格式为(0……31)
- %H：代表小时,格式为(00……23)
- %k：代表 小时,格式为(0……23)
- %h： 代表小时,格式为(01……12)
- %I： 代表小时,格式为(01……12)
- %l ： 代表小时,格式为(1……12)
- %i： 代表分钟, 格式为(00……59)
- %r：代表 时间,格式为12 小时(hh:mm:ss [AP]M)
- %T：代表 时间,格式为24 小时(hh:mm:ss)
- %S：代表 秒,格式为(00……59)
- %s：代表 秒,格式为(00……59)

Mysql 时间字符转换

- select date_format(now(),'%Y');
- select date_format(now(),'%Y-%c-%d %h:%i:%s');
- SELECT STR_TO_DATE('Jul 20 2013 7:49:14:610AM','%b %d %Y %h:%i:%s:%f%p') from DUAL;
- -- 执行后得到结果:
- '2013-07-20 07:49:14.610000'
- http://dev.mysql.com/doc/refman/5.7/en/date-and-time-functions.html#function_str-to-date
- http://dev.mysql.com/doc/refman/5.7/en/date-and-time-functions.html#function_date-format
- http://dev.mysql.com/doc/refman/5.7/en/