第6章 综合案例练习之基础查询

通过前面几章内容的学习，相信读者对Hive的具体使用已经有了一定的了解。本章将基于第3、4、5章讲解的Hive SQL语法，给出不同角度的综合案例练习题。首先是环境准备部分，读者可以跟随内容创建数据表并载入数据，供后续练习使用。接下来的练习题部分，读者可以先自行尝试解答，再对比给出的解答思路。需要读者注意的是，每道练习题的解答思路都不是唯一的，书中仅给出了其中一种解答思路，读者可以自行思考多种解决方案，并评估不同方案的性能，这也是后续内容将会重点讲解的。

6.1 环境准备

本章所有的练习题都将基于同一组数据表，本节需要完成数据准备工作

1. 创建数据表

hive>

-- 创建学生表

DROP TABLE IF EXISTS student\_info;

create table if not exists student\_info(

stu\_id string COMMENT '学生id',

stu\_name string COMMENT '学生姓名',

birthday string COMMENT '出生日期',

sex string COMMENT '性别'

)

row format delimited fields terminated by ','

stored as textfile;

-- 创建课程表

DROP TABLE IF EXISTS course\_info;

create table if not exists course\_info(

course\_id string COMMENT '课程id',

course\_name string COMMENT '课程名',

tea\_id string COMMENT '任课老师id'

)

row format delimited fields terminated by ','

stored as textfile;

-- 创建老师表

DROP TABLE IF EXISTS teacher\_info;

create table if not exists teacher\_info(

tea\_id string COMMENT '老师id',

tea\_name string COMMENT '老师姓名'

)

row format delimited fields terminated by ','

stored as textfile;

-- 创建分数表

DROP TABLE IF EXISTS score\_info;

create table if not exists score\_info(

stu\_id string COMMENT '学生id',

course\_id string COMMENT '课程id',

score int COMMENT '成绩'

)

row format delimited fields terminated by ','

stored as textfile;

2. 数据准备

（1）创建/opt/module/data目录

[atguigu@hadoop102 module]$ mkdir data

（2）将student\_info.txt、course\_info.txt、teacher\_info.txt和score\_info.txt 4个文件上传至/opt/module/data目录下。以上4个数据文件在本书附赠的资料中可以找到。

（3）4个文件中的数据内容分别如下所示，篇幅所限，仅展示部分数据。

[atguigu@hadoop102 data]$ vim student\_info.txt

001,陈富贵,1995-05-16,男

002,李建国,1994-03-20,男

003,杨建军,1995-04-30,男

004,刘爱党,1998-08-28,男

[atguigu@hadoop102 data]$ vim course\_info.txt

01,语文,1003

02,数学,1001

03,英语,1004

04,体育,1002

05,音乐,1002

[atguigu@hadoop102 data]$ vim teacher\_info.txt

1001,张高数

1002,李体音

1003,王子文

1004,刘丽英

[atguigu@hadoop102 data]$ vim score\_info.txt

001,01,94

002,01,74

004,01,85

005,01,64

3. 加载数据

（1）将数据文件分别加载至表中。

hive>

load data local inpath '/opt/module/data/student\_info.txt' into table student\_info;

load data local inpath '/opt/module/data/course\_info.txt' into table course\_info;

load data local inpath '/opt/module/data/teacher\_info.txt' into table teacher\_info;

load data local inpath '/opt/module/data/score\_info.txt' into table score\_info;

（2）验证插入数据情况

hive>

select \* from student\_info limit 5;

select \* from course\_info limit 5;

select \* from teacher\_info limit 5;

select \* from score\_info limit 5;

6.2 简单查询练习

1. 查询姓名中带“山”的学生名单

（1）思路分析。

本题主要考察对where子句与关系运算符like的结合使用。

（2）查询语句。

hive>

select

\*

from student\_info

where stu\_name like "%山%";

（3）查询结果。

stu\_id stu\_name birthday sex

006 廖景山 1992-11-12 男

010 吴山 1998-08-23 男

2. 查询姓“王”老师的个数

（1）思路分析。

本题主要考察对where子句与关系运算符like的结合使用，然后再结合count()聚合函数完成对符合结果的数据个数的统计。

（2）查询语句。

hive>

select

count(\*) wang\_count

from teacher\_info

where tea\_name like '王%';

（3）查询结果。

wang\_count

1

3. 检索课程编号为“04”且分数小于60的学生的分数信息，结果按分数降序排列

（1）思路分析。

本题主要通过where子句、关系运算符和逻辑运算符的综合使用得到课程编号为“04”且分数小于60的学生的课程信息，最后使用order by关键字进行排序操作。

（2）查询语句。

hive>

select

stu\_id,

course\_id,

score

from score\_info

where course\_id ='04' and score<60

order by score desc;

（3）查询结果。

stu\_id course\_id score

004 04 59

001 04 54

020 04 50

014 04 40

017 04 34

010 04 34

4. 查询数学成绩不及格的学生信息和其对应的数学学科成绩，按照学号升序排序

（1）思路分析。

本案例分两步完成。

第一步：根据学科名称为“数学”作为条件查询，找到数学学科对应的学科编号，再根据学科编号和分数小于60的条件，在score\_info表中获取到相关的成绩信息。

第二步：第一步已经获取到相关成绩信息，但是要求获取对应的学生信息，所以要用第一步中获取的成绩信息和学生表student\_info进行join关联查询，关联字段为学生编号stu\_id。最终获取到学生以及相关成绩信息，再使用order by关键字进行排序即可。

（2）查询语句。

hive>

select

s.stu\_id,

s.stu\_name,

t1.score

from student\_info s

join (

select

\*

from score\_info

where course\_id=(select course\_id from course\_info where course\_name='数学') and score < 60

) t1 on s.stu\_id = t1.stu\_id

order by s.stu\_id;

（3）查询结果。

s.stu\_id s.stu\_name t1.score

005 韩华翰 44

007 孟海 55

008 宋忠 34

011 邱钢 49

013 许晗晗 35

014 谢思萌 39

015 乔白凝 48

017 熊巧 34

018 黄瑗 58

019 乔颜 39

020 于丝 59

6.3 分组与汇总练习

6.3.1 汇总练习

1. 查询编号为“02”的课程的总成绩

（1）思路分析。

本题主要考查分组聚合知识点。

首先将score\_info表中课程编号为“02”的数据过滤出来，接下来为了统计该课程的总成绩，可以利用course\_id字段分组，再结合sum()聚合函数累加求和，最终获取成绩的总和。

（2）查询语句。

hive>

select

course\_id,

sum(score) score\_sum

from score\_info

where course\_id='02'

group by course\_id;

（3）查询结果。

course\_id score\_sum

02 1133

2. 查询参加考试的学生个数

（1）思路分析。

本题主要考查distinct去重，结合count聚合函数统计学生个数。

（2）查询语句。

hive>

select

count(distinct stu\_id) stu\_num

from score\_info;

（3）查询结果。

stu\_num

19

6.3.2 分组练习

1. 查询各科成绩最高和最低的分，以如下的形式显示：课程号、最高分、最低分

（1）思路分析。

本题主要考查了使用分组聚合求最大值和最小值。

按照course\_id字段进行分组，通过max和min函数获取各科成绩的最大值和最小值。

（2）查询语句。

hive>

select

course\_id,

max(score) max\_score,

min(score) min\_score

from score\_info

group by course\_id;

（3）查询结果。

course\_id max\_score min\_score

01 94 38

02 93 34

03 99 32

04 100 34

05 87 59

2. 查询每门课程有多少学生参加了考试（有考试成绩）

（1）思路分析。

本题主要考查分组聚合的使用。

根据题意，首先按照course\_id字段进行分组，再通过count函数对stu\_id进行个数统计，即可获取每一门课程的参加考试的学生人数。

（2）查询语句。

hive>

select

course\_id,

count(stu\_id) stu\_num

from score\_info

group by course\_id;

（3）查询结果。

course\_id stu\_num

01 19

02 19

03 19

04 12

05 5

3. 查询男生、女生人数

（1）思路分析。

本题主要考查的是分组聚合的使用。按照sex字段分组，并使用count聚合函数统计人数即可。

（2）查询语句。

hive>

select

sex,

count(stu\_id) count

from student\_info

group by sex;

（3）查询结果。

sex count

女 9

男 11

6.3.3 对分组结果的条件查询

1. 查询平均成绩大于60分的学生的学号和平均成绩

（1）思路分析。

首先使用分组聚合得到每个学生的平均成绩。然后对平均成绩按照指定条件过滤，得到平均成绩高于60分的学生和成绩。平均成绩是聚合得到的结果，对聚合结果的过滤需要使用having关键字，而不是where子句。

（2）查询语句。

hive>

select

stu\_id,

avg(score) score\_avg

from score\_info

group by stu\_id

having score\_avg > 60;

（3）查询结果。

stu\_id score\_avg

001 72.5

002 86.25

004 81.5

005 75.4

006 73.33333333333333

009 74.2

013 61.0

015 70.25

016 81.25

020 69.75

2. 查询至少考了四门课程的学生学号

（1）思路分析。

首先使用分组聚合，按照stu\_id进行分组，使用count聚合函数统计course\_id个数，得到每个学生考试课程的个数。最后使用having关键字对聚合后得到的课程个数进行过滤，得到选修课程数大于等于4的学生。

（2）查询语句。

hive>

select

stu\_id,

count(course\_id) course\_count

from score\_info

group by stu\_id

having course\_count >=4;

（3）查询结果。

stu\_id course\_num

001 4

002 4

004 4

005 5

007 5

009 5

010 4

013 4

014 4

015 4

016 4

017 4

018 4

020 4

3. 查询每门课程的平均成绩，结果按平均成绩升序排序，平均成绩相同时，按课程号降序排列

（1）思路分析。

本题主要思路是使用分组聚合计算每门课程的平均成绩，然后使用order by关键字按照题目要求对结果进行排序。使用order by关键字时，配合asc是升序，配合desc是降序，默认为升序。

（2）查询语句。

hive>

select

course\_id,

avg(score) score\_avg

from score\_info

group by course\_id

order by score\_avg asc, course\_id desc;

（3）查询结果。

course\_id score\_avg

02 59.63157894736842

04 63.416666666666664

01 67.15789473684211

03 69.42105263157895

05 74.6

4. 统计参加考试人数大于等于15的学科

（1）思路分析。

使用分组聚合得到每门学科参加考试的人数，使用having关键字过滤人数大于等于15的学科。

（2）查询语句。

hive>

select

course\_id,

count(stu\_id) stu\_count

from score\_info

group by course\_id

having stu\_count >= 15;

（3）查询结果。

course\_id stu\_count

01 19

02 19

03 19

6.3.4 查询结果排序和分组指定条件

1. 查询学生的总成绩并按照总成绩降序排序

（1）思路分析。

本题主要考查分组聚合和order by关键字的使用。

（2）查询语句。

hive>

select

stu\_id,

sum(score) sum\_score

from score\_info

group by stu\_id

order by sum\_score desc;

（3）查询结果。

stu\_id sum\_score

005 377

009 371

002 345

004 326

016 325

007 299

001 290

015 281

020 279

013 244

010 233

018 232

006 220

014 192

017 181

012 180

011 180

019 178

008 129

2. 查询一共参加三门课程且其中一门为语文课程的学生的id和姓名

（1）思路分析。

本题主要考查分组group by、使用having关键字进行条件过滤，以及多表关联的综合使用。

第一步： 在course\_info表中通过course\_name为“语文”查询得到语文课程的course\_id。

第二步：在score\_info表中查询所有学习语文课程的学生的stu\_id。

第三步：利用in关键字结合第二步的查询结果，查询成绩表score\_info，得到参加语文课程的学生的所有课程course\_id。

第四步：对第三步的查询结果按照stu\_id分组，使用count聚合函数得到学生参加的课程数，结合having关键字，筛选聚合结果为3的stu\_id。

第五步：将第四步的查询结果与student\_info表关联，得到学生姓名stu\_name字段。

（2）查询语句。

hive>

select

si.stu\_id,

si.stu\_name

from student\_info si

join

(

select stu\_id,

count(\*) cc

from score\_info

where stu\_id in (select stu\_id

from score\_info

where course\_id = (select course\_id from course\_info where course\_name = '语文'))

group by stu\_id

having count(\*) = 3

) t1

on

si.stu\_id=t1.stu\_id

（3）查询结果。

stu\_id stu\_name

006 廖景山

008 宋忠

011 邱钢

012 邓夏波

019 乔颜

6.4 复杂查询练习

1. 查询没有学全所有课的学生的学号、姓名

（1）思路分析。

对题目进行分析，没有学全所有课程，也就是说该学生选修的课程数量小于总的课程数量。

本题主要考查分组聚合与多表关联的综合使用。

第一步：将学生表作为主表和成绩表进行left join，并且按照学生学号和姓名分组 对每一个学生所学的课程进行count统计。

第二步：直接通过having的方式进行对没有学全的学生进行过滤。

（2）查询语句。

hive>

select sti.stu\_id,

sti.stu\_name,

count(sci.course\_id)

from student\_info sti

left join

score\_info sci

on

sti.stu\_id = sci.stu\_id

group by sti.stu\_id, sti.stu\_name

having count(sci.course\_id)<(select count(\*) from course\_info)

（3）查询结果。

t2.stu\_id t2.stu\_name t2.sc\_count

001 陈富贵 4

002 李建国 4

003 杨建军 0

004 刘爱党 4

006 廖景山 3

008 宋忠 3

010 吴山 4

011 邱钢 3

012 邓夏波 3

013 许晗晗 4

014 谢思萌 4

015 乔白凝 4

016 钟紫 4

017 熊巧 4

018 黄瑗 4

019 乔颜 3

020 于丝 4

2. 查询出只选修了三门课程的全部学生的学号和姓名

（1）思路分析。

本题主要考查分组后的聚合结果过滤，以及关联查询的使用。

第一步：查询成绩表，按照学生学号分组，然后使用count聚合函数对课程数进行统计，然后利用having关键字对数据进行过滤，得到选修课程数为3的结果。将查询结果作为临时表t1。

第二步：对学生表和临时表t1进行关联查询，关联字段为学生学号，获取学生姓名信息。

（2）查询语句。

hive>

select

s.stu\_id,

s.stu\_name

from student\_info s

join (

select

stu\_id,

count(course\_id) course\_count

from score\_info

group by stu\_id

having course\_count =3

) t1

on s.stu\_id = t1.stu\_id;

（3）查询结果。

s.stu\_id s.stu\_name

006 廖景山

008 宋忠

011 邱钢

012 邓夏波

019 乔颜

6.5 多表查询练习

6.5.1 表连接

1. 查询所有学生的学号、姓名、选课数、总成绩

（1）思路分析。

本题主要考查多表的关联查询和分组聚合的使用。

以学生表作为主表 使用left join和成绩表进行关联查询，stu\_id作为关联字段，再按照stu\_id和stu\_name作为分组字段，最终对课程数和总成绩进行统计。

（2）查询语句。

hive>

select

s.stu\_id,

s.stu\_name,

count(sc.course\_id) count\_course,

nvl(sum(sc.score),0) sum\_score

from student\_info s

left join score\_info sc on s.stu\_id = sc.stu\_id

group by s.stu\_id,s.stu\_name;

（3）查询结果。

stu\_id stu\_name course\_count course\_sum

001 陈富贵 4 290

002 李建国 4 345

003 杨建军 0 0

004 刘爱党 4 326

005 韩华翰 5 377

006 廖景山 3 220

007 孟海 5 299

008 宋忠 3 129

009 韩福 5 371

010 吴山 4 233

011 邱钢 3 180

012 邓夏波 3 180

013 许晗晗 4 244

014 谢思萌 4 192

015 乔白凝 4 281

016 钟紫 4 325

017 熊巧 4 181

018 黄瑗 4 232

019 乔颜 3 178

020 于丝 4 279

2. 查询平均成绩大于85的所有学生的学号、姓名和平均成绩

（1）思路分析。

本题依然主要考查多表的关联查询和分组聚合的使用。

以成绩表关联学生表，根据学生学号和学生名称作为分组字段，通过avg()函数获取组内平均成绩，最后使用having关键字过滤聚合结果，得到平均成绩大于85的数据。

（2）查询语句。

hive>

select s.stu\_id,

s.stu\_name,

avg(sc.score) avg\_score

from score\_info sc

join student\_info s on s.stu\_id = sc.stu\_id

group by s.stu\_id, s.stu\_name

having avg\_score > 85

（3）查询结果。

stu\_id stu\_name avg\_score

002 李建国 86.25

3. 查询学生的选课情况：学号，姓名，课程号，课程名称

（1）思路分析。

本题主要考查多表关联。

根据题目中的信息，需要分别关注成绩表、学生表、课程表，本题关键就是定位表与表之间的关联字段。course\_info表和score\_info表通过course\_id字段进行关联，student\_info表和score\_info表通过stu\_id字段进行关联。

（2）查询语句。

hive>

select

sti.stu\_id,

sti.stu\_name,

ci.course\_id,

ci.course\_name

from

student\_info sti

left join

score\_info sci

on

sti.stu\_id=sci.stu\_id

left join

course\_info ci

on

sci.course\_id=ci.course\_id

（3）查询结果。

sti.stu\_id sti.stu\_name ci.course\_id ci.course\_name

001 陈富贵 01 语文

001 陈富贵 02 数学

001 陈富贵 03 英语

001 陈富贵 04 体育

002 李建国 01 语文

002 李建国 02 数学

002 李建国 03 英语

002 李建国 04 体育

003 杨建军 NULL NULL

004 刘爱党 01 语文

004 刘爱党 02 数学

004 刘爱党 03 英语

004 刘爱党 04 体育

.....

答案一共75行 只做部分展示

4. 查询课程编号为03且课程成绩在80分以上的学生的学号和姓名及课程信息

（1）思路分析。

本题主要考查多表关联和条件过滤的综合使用。

第一步：查询成绩表，过滤课程编号为03且课程成绩在80分以上的信息，结果作为临时表t1。

第二步：查询学生表并join临时表t1，关联字段为stu\_id，同时join课程表，关联字段为course\_id，获取最终结果。

（2）查询语句。

hive>

select

s.stu\_id,

s.stu\_name,

t1.score,

t1.course\_id,

c.course\_name

from student\_info s

join (

select

stu\_id,

score,

course\_id

from score\_info

where score > 80 and course\_id = '03'

) t1

on s.stu\_id = t1.stu\_id

join course\_info c on c.course\_id = t1.course\_id;

（3）查询结果。

s.stu\_id s.stu\_name t1.score t1.course\_id c.course\_name

002 李建国 87 03 英语

004 刘爱党 89 03 英语

005 韩华翰 99 03 英语

013 许晗晗 93 03 英语

015 乔白凝 84 03 英语

019 乔颜 93 03 英语

020 于丝 81 03 英语

Time taken: 9.064 seconds, Fetched: 7 row(s)

6.5.2 多表连接

1. 课程编号为"01"且课程分数小于60，按分数降序排列的学生信息

（1）思路分析。

第一步：查询成绩表，过滤出课程编号为“01”且课程分数小于60的信息，结果作为临时表t1。

第二步：查询学生表join关联临时表t1，关联字段为stu\_id，获取相关学生信息，最后对成绩字段进行倒序排列。

（2）查询语句。

hive>

select

s.stu\_id,

s.stu\_name,

s.birthday,

s.sex,

t1.score

from student\_info s

join (

select

stu\_id,

course\_id,

score

from score\_info

where score < 60 and course\_id = '01'

) t1

on s.stu\_id=t1.stu\_id

order by t1.score desc;

（3）查询结果。

s.stu\_id s.stu\_name s.birthday s.sex t1.score

017 熊巧 1992-07-04 女 58

008 宋忠 1994-02-06 男 56

007 孟海 1999-04-09 男 48

013 许晗晗 1997-11-08 女 47

019 乔颜 1994-08-31 女 46

012 邓夏波 1996-12-21 女 44

018 黄瑗 1993-09-24 女 38

Time taken: 8.936 seconds, Fetched: 7 row(s)

2. 查询所有课程成绩在70分以上的学生的姓名、课程名称和分数，按分数升序排列

（1）思路分析。

按照题目要求，本题的关键在于如何查询得到所有课程成绩在70分以上的的学生信息。

第一步：首先需要按照学生学号分组，将sum聚合函数与if逻辑判断函数巧妙联合使用，课程成绩在70分以上计为0，否则为1，再将其sum求和作为flage字段，使用having关键字过滤flage，flage为0的即为所有课程成绩在70分以上的学生学号，作为临时表t1。

第二步：查询学生表，通过join和临时表t1关联，关联字段为stu\_id，获取学生姓名息。

第三步：与成绩表和课程表做join关联查询，获得课程名和成绩信息。

（2）查询语句。

hive>

select

s.stu\_id,

s.stu\_name,

c.course\_name,

s2.score

from student\_info s

join (

select

stu\_id,

sum(if(score >= 70,0,1)) flage

from score\_info

group by stu\_id

having flage =0

) t1

on s.stu\_id = t1.stu\_id

join score\_info s2 on s.stu\_id = s2.stu\_id

join course\_info c on s2.course\_id = c.course\_id

order by S2.score

（3）查询结果。

s.stu\_id s.stu\_name c.course\_name s2.course

016 钟紫 语文 71

016 钟紫 英语 71

002 李建国 语文 74

002 李建国 数学 84

002 李建国 英语 87

016 钟紫 数学 89

016 钟紫 体育 94

002 李建国 体育 100

Time taken: 27.166 seconds, Fetched: 8 row(s)

3. 查询该学生不同课程的成绩相同的学生编号、课程编号、学生成绩

（1）思路分析。

本题主要考查表关联时关联条件的灵活应用。

此题主要针对成绩表进行查询，关键使用成绩表和自身进行关联来获取数据，关联条件就是学生编号相同、课程编号不同，以及成绩相同。简单来说就是查询一个学生哪些不同课程考了相同的成绩。

（2）查询语句。

hive>

select

sc1.stu\_id,

sc1.course\_id,

sc1.score

from score\_info sc1

join score\_info sc2 on sc1.stu\_id = sc2.stu\_id

and sc1.course\_id <> sc2.course\_id

and sc1.score = sc2.score;

（3）查询结果。

sc1.stu\_id sc1.course\_id sc1.score

016 03 71

017 04 34

016 01 71

005 05 85

007 05 63

009 05 79

017 02 34

005 04 85

007 04 63

009 04 79

Time taken: 8.881 seconds, Fetched: 10 row(s)

4. 查询课程编号为“01”的课程比“02”的课程成绩高的所有学生的学号

（1）思路分析。

本题主要考查嵌套查询结合join关联查询获取信息。

第一步：首先查询成绩表，将课程编号为01和课程编号为02的信息获取到，结果分别为临时表s1和s2。

第二步：将临时表s1和临时表s2进行join关联，关联字段为学生编号stu\_id，并结合s1中的成绩大于s2的成绩为条件进行过滤获取结果。

（2）查询语句。

hive>

select

s1.stu\_id

from

(

select

sc1.stu\_id,

sc1.course\_id,

sc1.score

from score\_info sc1

where sc1.course\_id ='01'

) s1

join

(

select

sc2.stu\_id,

sc2.course\_id,

score

from score\_info sc2

where sc2.course\_id ="02"

)s2

on s1.stu\_id=s2.stu\_id

where s1.score > s2.score;

（3）查询结果。

stu\_id

001

005

008

010

011

013

014

015

017

019

020

5. 查询学过编号为“01”的课程并且也学过编号为“02”的课程的学生的学号、姓名

（1）思路分析。

本题主要考查嵌套查询结合join关联获取信息。

第一步：查询成绩表，获取出课程编号为01的学生编号，再次查询成绩表将课程编号为01的学生编号作为联合条件过滤出同时也选修了课程编号为02的课程的学生编号，查询结果为临时表t1。

第二步：将临时表t1和学生表student\_info进行join关联学生姓名。

（2）查询语句。

hive>

select

t1.stu\_id as `学号`,

s.stu\_name as `姓名`

from

(

select

stu\_id

from score\_info sc1

where sc1.course\_id='01'

and stu\_id in (

select

stu\_id

from score\_info sc2

where sc2.course\_id='02'

)

)t1

join student\_info s

on t1.stu\_id = s.stu\_id;

（3）查询结果。

学号 姓名

001 陈富贵

002 李建国

004 刘爱党

005 韩华翰

006 廖景山

007 孟海

008 宋忠

009 韩福

010 吴山

011 邱钢

012 邓夏波

013 许晗晗

014 谢思萌

015 乔白凝

016 钟紫

017 熊巧

018 黄瑗

019 乔颜

020 于丝

Time taken: 10.161 seconds, Fetched: 19 row(s)

6. 查询学过“李体音”老师所教的所有课的同学的学号、姓名

（1）思路分析。

第一步：根据题目要求，通过将课程表和老师表关联获取“李体音”老师所教课程的课程编号。

第二步：将第一步中获取的课程编号作为条件，查询成绩表获取相关信息，并根据学生编号分组，通过having关键字组内进行过滤，过滤条件很关键，要想获得学过“李体音”老师所教的所有课的同学，必须满足count统计组内条数等于李体音的课程总数的条件。将查询结果作为临时表t1。

第三步：将临时表t1和学生表进行关联查询，获取最终的结果。

（2）查询语句。

hive>

select t1.stu\_id,

si.stu\_name

from (

select stu\_id

from score\_info si

where course\_id in

(

select course\_id

from course\_info c

join teacher\_info t

on c.tea\_id = t.tea\_id

where tea\_name = '李体音'

)

group by stu\_id

having count(\*) = (select count(\*)

from course\_info c

join teacher\_info t

on c.tea\_id = t.tea\_id

where tea\_name = '李体音')

) t1

join student\_info si

on t1.stu\_id = si.stu\_id;

（3）查询结果。

s.stu\_id s.stu\_name

005 韩华翰

007 孟海

009 韩福

Time taken: 27.16 seconds, Fetched: 3 row(s)

7. 查询学过“李体音”老师所讲授的任意一门课程的学生的学号、姓名

（1）思路分析。

本题和上一题的要求相近，区别在于是学过“李体音”老师的所有课还是任意一门。所以在第二步中，不再对查询结果做有条件过滤。

第一步：根据题目要求，通过将课程表和老师表关联获取“李体音”老师所教课程的课程编号。

第二步：将第一步中获取的课程编号作为条件，查询成绩表获取相关信息，并根据学生编号分组，将查询结果作为临时表t1。

第三步：将临时表t1和学生表进行关联查询，获取最终的结果。

（2）查询语句。

hive>

select

t1.stu\_id,

si.stu\_name

from

(

select

stu\_id

from score\_info si

where course\_id in

(

select

course\_id

from course\_info c

join teacher\_info t

on c.tea\_id = t.tea\_id

where tea\_name='李体音'

)

group by stu\_id

)t1

join student\_info si

on t1.stu\_id=si.stu\_id;

（3）查询结果。

s.stu\_id s.stu\_name

001 陈富贵

002 李建国

004 刘爱党

005 韩华翰

007 孟海

009 韩福

010 吴山

013 许晗晗

014 谢思萌

015 乔白凝

016 钟紫

017 熊巧

018 黄瑗

020 于丝

Time taken: 9.391 seconds, Fetched: 14 row(s)

8. 查询没学过"李体音"老师讲授的任一门课程的学生姓名

（1）思路分析。

本题考查的是上一题的查询结果的反向查询结果，关键在于not in关键字的使用。

第一步：根据题目要求，通过将课程表和老师表关联获取“李体音”老师所教课程的课程编号。

第二步：将第一步中获取的课程编号作为条件，查询成绩表做in包含查询，获取“李体音”老师所教的学生编号，并根据学生编号分组。

第三步：查询学生信息表，根据第二步查询的结果做not in过滤，获取结果。

（2）查询语句。

hive>

select

stu\_id,

stu\_name

from student\_info

where stu\_id not in

(

select

stu\_id

from score\_info si

where course\_id in

(

select

course\_id

from course\_info c

join teacher\_info t

on c.tea\_id = t.tea\_id

where tea\_name='李体音'

)

group by stu\_id

);

（3）查询结果。

stu\_id stu\_name

003 杨建军

006 廖景山

008 宋忠

011 邱钢

012 邓夏波

019 乔颜

Time taken: 36.559 seconds, Fetched: 6 row(s)

9. 查询至少有一门课与学号为“001”的学生所学课程相同的学生的学号和姓名

（1）思路分析。

本题主要考查多条件数据过滤，关键是分析需求利用逆向思维解题。

第一步：查询成绩表中学生编号为001的课程编号。

第二步：将成绩表和学生表进行关联查询，关联字段为学生编号，并按照学生编号和姓名分组。根据第一步获取的课程编号作为条件做in包含查询。这里还有关键一步就是查询条件中要对001自身进行排除。

（2）查询语句。

hive>

select

si.stu\_id,

si.stu\_name

from score\_info sc

join student\_info si

on sc.stu\_id = si.stu\_id

where sc.course\_id in

(

select

course\_id

from score\_info

where stu\_id='001' --001的课程

) and sc.stu\_id <> '001' --排除001学生

group by si.stu\_id,si.stu\_name;

（3）查询结果。

s1.stu\_id s2.stu\_name

002 李建国

004 刘爱党

005 韩华翰

006 廖景山

007 孟海

008 宋忠

009 韩福

010 吴山

011 邱钢

012 邓夏波

013 许晗晗

014 谢思萌

015 乔白凝

016 钟紫

017 熊巧

018 黄瑗

019 乔颜

020 于丝

Time taken: 8.97 seconds, Fetched: 18 row(s)

10. 按平均成绩从高到低显示所有学生的所有课程的成绩以及平均成绩

（1）思路分析。

本题主要考查多表联查结合分组统计查询。

第一步：查询成绩表，按照学生编号分组，获取每一个学生的平均成绩。作为临时结果表t1。

第二步：查询成绩表，并join关联学生表和课程表，以及临时结果表t1，获取综合信息，最后按照t1中的平均成绩进行order by desc倒序排列。

（2）查询语句。

hive>

select

si.stu\_name,

ci.course\_name,

sc.score,

t1.avg\_score

from student\_info si

left join score\_info sc

on sc.stu\_id=si.stu\_id

left join course\_info ci

on sc.course\_id=ci.course\_id

left join

(

select

stu\_id,

avg(score) avg\_score

from score\_info

group by stu\_id

)t1

on sc.stu\_id=t1.stu\_id

order by t1.avg\_score desc;

（3）查询结果。

t2.stu\_name t2.course\_name t2.score t1.avg\_score

李建国 数学 84 86.25

李建国 英语 87 86.25

李建国 体育 100 86.25

李建国 语文 74 86.25

刘爱党 体育 59 81.5

.....

熊巧 体育 34 45.25

熊巧 英语 55 45.25

熊巧 数学 34 45.25

熊巧 语文 58 45.25

宋忠 英语 39 43.0

宋忠 语文 56 43.0

宋忠 数学 34 43.0

杨建军 NULL NULL NULL

Time taken: 20.137 seconds, Fetched: 75 row(s)

6.6 本章总结

本章的主要内容是结合前面章节讲解的基础查询语法给出的综合案例练习题，主要考察单表和多表的关联查询中，常用关键字的综合使用，其中包含了对分组聚合的大量练习。本章通过大量的基础练习案例，呈现了Hive SQL语法的基本知识点，并锻炼了读者对业务需求分析的基本思路，旨在使Hive初学者更快更好入门，为后面更复杂的Hive SQL使用和练习做准备。