第3章

SQL Server 数据定义与操作

关系数据库的标准语言是 SQL(Structured Query Language,结构化查询语言),虽然 SQL 字面含义是“查询语言”,但其功能却包括数据定义、查询、更新和保护等许多内容。实际中,各种不同数据库管理系统对 SQL 的实现都存有小的差异。以 SQL Server 为例,本章讲述基本数据定义和常用查询及更新,第 4 章讲述应用环境中的 SQL Server,第 5 章讲述数据保护。本章结构安排如下:3.1 节是 SQL 与 SQL Server 概述,3.2 节介绍数据定义,3.3 节～3.11 节介绍数据查询,3.12 节介绍数据更新。

3.1 SQL 与 SQL Server

3.1.1 SQL 发展史

1970 年,美国 IBM 研究中心的 E. F. codd 连续发表多篇论文,提出关系模型。1972 年,IBM 公司开始研制实验型关系数据库管理系统 SYSTEM R,配制的查询语言称为 SQUARE(Specifying Queries As Relational Expression);1974 年,同一实验室的 R. F. Boyce 和 D. D. Chamberlin 对 SQUARE 进行改进,减少了一些数学符号,采用英语单词表示和结构式的语法规则,并重命名为 SEQUEL(Strutured English Query Language);后来 SEQUEL 简称为 SQL(Structured Query Language),即“结构化查询语言”。

1986 年,美国国家标准化协会(ANSI)发布了 ANSI 文件 X5.135—1986《数据库语言 SQL》,1987 年 6 月,国际标准化组织(ISO)采纳其为国际标准,现在称为“SQL86”。ANSI 在 1989 年 10 月又颁布了增强完整性特征的 SQL89 标准。随后,ISO 对标准进行了大量的修改和扩充,在 1992 年 8 月发布了标准化文件 ISO/IEC9075:1992《数据库语言 SQL》。人们习惯称这个标准为“SQL2”。1999 年,ISO 发布了标准化文件 ISO/IEC9075:1999《数据库语言 SQL》,人们习惯称这个标准为“SQL3”。最新发布的 SQL 标准是 SQL:2016,下一版本的发布工作已经在进行之中。实践中,各种数据库管理系统在其实现中都对 SQL 规范既兼容又扩充。在未来很长一段时间里,SQL 仍将是数据库领域的主流语言。不仅如此,像 SQL 一样简单是数据管理系统的普遍追求,万

维网联盟(W3C)发布的 XML 数据查询语言标准 XQuery 就模仿 SQL,RDF 数据查询语言标准 SPARQL 被称为 RDF 上的 SQL,大数据领域也已将提供类 SQL 查询语言作为大数据管理的重要技术目标之一,比如微软的 DryadLINQ 就使用类似 SQL 的非过程化声明式语言。

3.1.2 SQL Server

SQL Server 是 Microsoft 公司推出的关系型数据库管理系统,具有使用方便、可伸缩性好等优点,可跨越从运行 Microsoft Windows 98 的笔记本计算机到运行 Microsoft Windows 2012 的大型多处理器的服务器等多种平台使用。

Microsoft SQL Server 是一个全面的数据库平台,使用集成的商业智能(BI)工具提供了企业级的数据管理,其官方版不仅具有现有数据平台的强大能力,还能快速构建相应的解决方案实现私有云与公有云之间数据的扩展与应用的迁移,帮助数以千计的企业用户突破性地快速实现各种数据体验。

3.1.3 数据库语言组成

数据库语言基本部分主要有:①数据定义语言,即 SQL DDL,用来创建数据库中的各类对象,如表的创建、撤销与更改;②数据操作语言,即 SQL DML,数据操作分为数据查询和数据更新,用于实现投影、选择、笛卡儿积和连接、聚集查询和数据插入、删除和修改三种操作;③数据保护语言,即 SQL DPL,可用来授予或收回访问数据库的权限,协调事务间的动作等,由 DBMS 提供统一数据保护功能,维护数据的保密性、完整性和可用性,是数据库系统的主要特征之一。

过去,SQL 是面向非过程编程与操作,需要和其他语言结合使用。SQL Server 能和其他语言自然融合,如 C 语言,SQL Server 为其嵌入式 SQL 提供了一些特殊的接口,通过预处理程序将 SQL 转为 C 语句。

3.1.4 数据库语言特点

(1) SQL 是非过程化的第四代编程语言。用户无须指定对数据的存放方法,因此,其他语言需要一大段程序才能实现的功能,SQL 只需要一个语句就可以实现。SQL 具有十分灵活和强大的查询功能,其 SELECT 语句能完成相当复杂的查询操作。有人把非过程化的编程语言称为第四代语言,特点是只要提出“做什么”,而无须说明“怎么做”。

(2) 一种语言多种使用方式。SQL 是“自含式”语言,也是“嵌入式”语言。作为自含式语言,SQL 不仅可以作为程序语言进行编程使用,而且可以作为交互命令使用,用户可以在终端键盘上直接输入 SQL 命令对数据库进行操作。作为嵌入式语言,SQL 还可以嵌入到高级语言的程序中。在两种不同的使用方式下,SQL 的语法结构基本一致。

(3) SQL 是关系数据库的标准语言,作为数据存取共同标准接口,可使不同数据库连接成一个统一的整体、有利于各种数据库之间交换数据、程序的移植及实现高度的数据独立性。SQL 的影响已经超出了数据库领域本身,不少软件产品将 SQL 的数据查询功能与多媒体图形功能、软件工程工具、软件开发工具和人工智能程序相结合,显示出了相当

大的潜力，应用领域前途无量。近年来，随着 Internet 技术的迅猛发展和快速普及，人们在 HTML 和 XML 中加入 SQL 语句，通过 WWW 来访问数据库，使得 SQL 的应用更加广泛和深入。

(4) 结构简洁、易学易用。SQL 是面向数据的语言，它集数据定义、数据操作和数据保护等数据库必需的基本功能为一身，充分体现了关系数据库的本质特点和巨大优势。SQL 不但功能极强，而且设计构思非常巧妙，语言结构简洁明快，语句非常接近英语语句，特别易学易用。

3.1.5　考试系统数据库

考试系统包括一个题目数据库，里面有大量各个专业、各门课程、各类考试(包括考试、测验和练习)可用的题目，各个院系的考官每次可以从中选择合适的题目组成试卷，称为组卷，考生可以根据需要报考某份试卷，答卷后获得一个分数，称为答卷。考试系统数据库如下。

考生表：examinee(eeid，eename，eesex，eeage，eedepa)，如表 3-1 所示。

表 3-1　考生表 examinee

eeid	eename	eesex	eeage	eedepa
218811011013	刘诗诗	男	20	历史学院
218811011014	刘诗诗	男	21	历史学院
218811011219	王琳懿	女	18	文学院
218811011220	王琳懿	女	19	文学院
218811011221	刘慧杰	女	19	文学院
218811011117	刘慧杰	女	19	教育学部
218811011025	张立帆	男	20	心理学院
218811011027	张立帆	男	19	心理学院
218811011028	刘慧杰	男	20	心理学院

考官表：examiner(erid，ername，ersex，erage，ersalary，erdepa)，如表 3-2 所示。

表 3-2　考官表 examiner

erid	ername	ersex	erage	ersalary	erdepa
2009040	成志云	女	35	5000	历史学院
1990122	戴小刚	男	53	8000	教育学部
1998039	丁向军	女	42	6500	文学院
2011049	郑博宇	男	32	5000	物理系
2007033	李晓燕	女	38	5000	心理学院

续表

erid	ername	ersex	erage	ersalary	erdepa
1995057	林永强	男	49	6500	历史学院
2013069	王瑞芬	女	30	5000	心理学院
2010022	姚翠红	女	36	5000	物理系

试卷表：exampaper（eid，ename，etype，eduration），如表 3-3 所示。

表 3-3　试卷表 exampaper

eid	ename	etype	eduration
0205000002	中国近现代史纲要	4	100
0210000001	大学外语	2	180
0201020001	计算机应用基础	4	120
0211000001	大学美育	3	120
0219001014	普通物理学	4	100
0110001001	教育学	1	180
0110001002	心理学	1	180

考生答卷表：eeexam(eeid,eid,achieve)，如表 3-4 所示。

表 3-4　考生答卷表 eeexam

eeid	eid	achieve
218811011013	0205000002	92
218811011013	0210000001	85
218811011013	0201020001	88
218811011116	0210000001	90
218811011116	0201020001	80

考官制卷表：erexam(erid,eid)，如表 3-5 所示。

表 3-5　考官制卷表 erexam

erid	eid
2009040	0205000002
1998039	0211000001
2007033	0110001002
2010022	0219001014
1990122	0110001001

院系表：department(dname,dloca,dtele)，如表 3-6 所示。

表 3-6　院系表 department

dname	dloca	dtele
历史学院	主楼 B2	58809289
教育学部	英东教育楼	58808855
文学院	主楼 B7	58807998
物理系	物理楼	58808135
心理学院	后主楼 12	58807832

3.1.6　应急预案编制数据库

应急预案指面对突发事件如自然灾害、重特大事故、环境公害及人为破坏的应急管理、指挥、救援计划等，是一种公文。通常一个应急预案由多个不同的编制单位协同编写，才能编制完成。应急预案包含预案编号(plan_id)，预案名(plan_name)，针对的灾害类型(plan_disatype)，针对的区域(plan_area)，针对的灾害等级(plan_level)，发布时间(plan_date)。应急预案编制的参与单位包含单位编号(depart_id)，单位名称(depart_name)，单位联系方式(depart_tel)。一个参与单位可能参与多个预案的编制，一个预案需要多个参与单位协作完成。当参与单位完成编写应急预案时，会记录该单位在应急预案编制中的职责(depart_respon)和工作量(workload)。如表 3-7～表 3-12 所示。

表 3-7　参与单位表 departments 结构

属　性	类　型	长　度	是否为主键
depart_id	int		是
depart_name	varchar	50	否
depart_tel	varchar	30	否

表 3-8　departments 表中的数据

depart_id	depart_name	depart_tel
1	教育部	58789087
2	应急部	58789768
3	民政部	58786272
4	国防部	58784585

表 3-9　应急预案表 plan 的结构

属　性	类　型	长度	是否为主键
plan_id	int		是
plan_name	varchar	50	否

续表

属　性	类　型	长度	是否为主键
plan_disatype	varchar	20	否
plan_area	varchar	30	否
plan_level	varchar	10	否
plan_date	datetime		否

表 3-10　plan 表中数据

plan_id	plan_name	plan_disatype	plan_area	plan_level	plan_date
1	山东省台风应急预案	自然灾害	山东	三级	2019-08-21
2	江西省暴雨应急预案	自然灾害	江西	一级	2018-07-19
3	汶川地震应急预案	自然灾害	汶川	二级	2008-05-12

表 3-11　参与情况表 record 结构

属　性	类　型	长　度	是否为主键	是否外键
depart _id	int		是	是
plan_id	int		是	是
depart_respon	varchar	20	否	否
workload	varchar	30	否	否

表 3-12　参与情况表 record 中数据

depart_id	plan_id	depart_respon	workload
2	1	安置受灾群众	10 天
3	2	灾后重建	30 天
4	1	宣传教育	3 天

3.1.7　中共党史数据库

中共党史数据库事件表 event 包含中国共产党成立与发展过程中的重要事件，包含事件编号(event_id)、事件日期(event_date)、事件地点(event_place)、事件名称(event_name)，如表 3-13 所示，其中的示例数据如表 3-14 所示。

表 3-13　中共党史数据库事件表 event 结构

属　性	类　型	长　度	是否为主键
event _id	int		是
event_date	datetime		否

续表

属　性	类　型	长　度	是否为主键
event_place	varchar	20	否
event_name	varchar	50	否

表 3-14　中共党史数据库事件表 event 中数据

event_id	event_date	event_place	event_name
1	1919-05-04	北京	五四运动
2	1921-07-23	上海	中国共产党第一次全国代表大会
3	1978-12-18	北京	中国共产党第十一届中央委员会第三次全体会议

3.2　数 据 定 义

数据定义包括数据库对象的创建、删除和修改三个部分。SQL Server 中对象标识符和 SQL Server 保留字必须以一个字母(a～z 以及带变音符的字母和非拉丁字母)或下画线(_)开头,随后的字符可以是字母、下画线、数字(0～9)、美元符号($)。SQL Server 中大小写不敏感,只有引号里面的字符才区分大小写。SQL Server 建议 SQL 的保留字用大写;表名、属性名等所有数据库对象名全部使用小写(除非从命名到使用每次都加双引号,否则所有对象名自动转换为小写)。

表是数据库中最重要、最基本的操作对象,是数据存储的基本单位。数据在表中是按照行和列的格式来存储的。每行代表唯一一个元组,每列代表一个域。创建表就是约定表中各个属性的属性名及其数据类型(实际上还包括约束定义,将在第 4 章讲述)。

3.2.1　SQL Server 的基本数据类型

SQL Server 支持的数据类型主要有整数数据类型、浮点数据类型、字符数据类型等,SQL Server 中 SQL 支持的基本数据类型如表 3-15 所示。

表 3-15　SQL 中的基本数据类型

名　字	存 储 空 间	描　述
SMALLINT	2B	小整数
INT(INTEGER)	4B	普通整数
BIGINT	8B	大整数
TINYINT	1B	极小整数
REAL	4B	变精度,不精确,6 位十进制数字精度
VARCHAR(n)	变长	有长度限制

续表

名　字	存储空间	描　述
CHAR(*n*)	定长	不足补空白
BIT	1B	其值为 0 或 1
DATETIME	8B	日期 YYYY-MM-DD

在字符串类型数据中，CHAR(*n*)指长度固定的字符串，*n* 表示列长度，VARCHAR(n)指长度可变的字符串，*n* 表示最大列长度。

3.2.2 表的创建、修改和删除

对表结构的操作有创建、修改和删除三种操作。

1. 表的创建

创建表使用 CREATE TABLE 语句，该语句的一般格式为：

```
CREATE TABLE <表名>
(<列名><数据类型>[<默认值>]
[,<列名><数据类型>[<默认值>]]……
);
```

其中，[]内的“[……]”是可选项，<表名>是所要定义的表名称。

表中每个列的类型可以是基本数据类型，也可以是用户预先定义的域名。主键是一种最基本的完整性约束，完整性约束将在第 4 章详细介绍。下面举例说明。

[例 3-1] 对于考试系统数据库中的三个关系表：

```
试卷表: exampaper(eid,ename,etype)
考官表: examiner(erid,ername, ersex,erage, ersalary,erdepa)
考官制卷表: erexam(erid,eid)
```

表 examiner 可以用以下语句创建。

```
CREATE TABLE examiner
(
    erid VARCHAR(12),
    ername VARCHAR(20),
    ersex CHAR(2),
    erage SMALLINT,
    ersalary REAL,
    erdepa VARCHAR(20)
    );
```

对于表 erexam 可以用下列语句创建。

```
CREATE TABLE erexam
(
erid VARCHAR(12),
eid VARCHAR(10)
);
```

对于表 exampaper 可以用下列语句创建。

```
CREATE TABLE exampaper
(
    eid VARCHAR(10),
    ename VARCHAR(20),
    etype SMALLINT,
    eduration VARVHAR(10)
    );
```

由上述语句可以看到：

(1) 表的定义就是逐列说明属性名称和属性类型。

(2) 每个语句末尾用分号“;”,表示语句结束。

(3) 用 CREATE 语句创建的表,最初只是一个空的框架,接下来,可用 INSERT 命令把数据插入表中(见 3.12 节)。另外,关系数据库产品都有数据装载程序,可以把大量原始数据导入表中。

2. 表结构的修改

在表建立并使用一段时期后,可能需要对表的结构进行修改,如增加新的属性、删除原有的属性或修改数据类型、宽度等。SQL 使用 ALTER TABLE 语句进行表结构修改。在进行表更新时,如果是新增属性,需要新属性一律为空值;如果是修改原有属性,则要注意是否可能破坏已有数据。

(1) 增加新的属性用“ALTER…ADD …”语句,其句法如下。

```
ALTER TABLE <表名>ADD <列名><类型>
```

[例 3-2]　在表 event 中增加一个事件描述(event_describe)列,可用下列语句。

```
ALTER TABLE event ADD event_describe VARCHAR(50);
```

表在增加一列后,原有元组在新增加的列上的值都被定义为空值(NULL)。

(2) 删除原有的属性用“ALTER…DROP COLUMN…”语句,其句法如下。

```
ALTER TABLE <表名>DROP COLUMN<列名>
```

[例 3-3]　在表 event 中删除事件描述(event_describe)列,可用下列语句。

```
ALTER TABLE event DROP COLUMN event_describe;
```

(3) 修改属性的数据类型,使用下面的命令:

```
ALTER TABLE … ALTER COLUMN…;
```

[例 3-4] 把表 event 中事件编号 event_id (int) 改为 varchar(10)可用下列语句。

```
ALTER TABLE event ALTER COLUMN event_id VARCHAR(10);
```

3. 表的撤销

在表不需要时,可以用 DROP TABLE 语句撤销。在一个表撤销后,其所有数据也就丢失了。

撤销语句的句法如下。

```
DROP  TABLE <表名>
```

[例 3-5] 撤销表 event,可用下列语句实现。

```
DROP TABLE event;
```

3.3 投影与广义投影

投影是指选取表中的某些属性的属性值;广义投影是对投影的扩展,指在选取属性列时,允许进行算术运算,即允许涉及属性和常量的算术表达式。投影与广义投影都用 SELECT 命令,语法如下。

```
SELECT [ALL|DISTINCT] <目标列表达式>[,<目标列表达式>] …
FROM <表名>[, <表名>] …
```

需要说明:

(1) 查询输入是 FROM 子句中列出的关系。

(2) 最简单的(<目标列表达式> [, <目标列表达式>] …)是 * ,它输出 FROM 子句中表的所有属性;否则,它是一个逗号分隔的表达式列表。表达式可能是一个属性名,也可以是任意常量算术表达式,如果是一般表达式,那么原理上向返回的表中增加一个新的虚拟属性。表达式为结果中的每一行进行一次计算,计算之前用该行的数值替换任何表达式里引用的属性。

(3) 投影结果中可能出现各个属性值均相等的元组对,但从数据库管理系统实现的角度看,投影过程会对每个新产生的结果元组进行标识,即能把每个元组视为不同。也就是说,由于去重是一项耗时的工作,DBMS 采取惰性原则:除非用户明确指出要求去重(即在 SELECT 保留字后跟 DISTINCT),否则认为每个元组皆不同。SELECT 保留字后

跟 ALL 指要求保留重复。

(4) 可以对结果表中行的显示次序进行控制。ORDER BY 子句让查询结果中的行按一个或多个属性(表达式)进行排序,升序时用 ASC,排序列为空值的行最后显示;降序时用 DESC,排序列为空值的行最先显示。

(5) 投影和广义投影都不会对原表产生任何改变,其他查询类似。

[例 3-6]　查询全部试卷的试卷号与试卷名。

试卷号和试卷名全部数据都存在表 exampaper 中,所以 FROM exampaper,需要列出每一个 eid 和每一个 ename 的值,所以 SELECT eid,ename,整个查询语句就是:

```
SELECT eid,ename
FROM exampaper;
```

这个语句的执行过程就是对 exampaper 表,输出其每一行的试卷号 eid 和试卷名 ename 值。

[例 3-7]　查询全部试卷号 eid 对应的组卷考官号,即输出所有试卷号、考官号。

试卷号和考官号的对应关系全部都存在表 erexam 中,所以 FROM erexam,需要列出每一个 eid 和每一个 erid 的值,所以 SELECT eid,erid,整个查询语句就是:

```
SELECT eid,erid
FROM erexam;
```

[例 3-8]　查询全部试卷本身的属性信息,也就是输出 exampaper 表中全部行的所有列。

试卷本身的属性信息都存在 exampaper 表中(注意:试卷报考信息存在 eeexam 表中,试卷组卷信息存在 erexam 表中,但是这些数据并不是查询要求的),所以 FROM exampaper,需要输出 exampaper 表每一行的所有属性列,可以用 SELECT * 或在 SELECT 后面列出该表的所有属性列名,完整的查询语句是:

```
SELECT eid, ename, etype, eduration
FROM exampaper;
```

或

```
SELECT *
FROM exampaper;
```

[例 3-9]　查询全部预案编制参与单位的信息。

```
SELECT *
FROM departments;
```

[例 3-10]　查询每位考生每门考试的扣分情况,即成绩与满分(100)之差。

```
SELECT eeid,100-achieve
FROM eeexam;
```

该查询结果如表 3-16 所示。

表 3-16 查询结果

eeid	100-achieve
218811011013	8
218811011013	15
218811011013	12
218811011116	10
218811011116	20

[例 3-11] 查询全体考官情况，查询结果按所在院系名升序排列，同一学院中的考官按年龄降序排列。

```
SELECT *
FROM examiner
ORDER BY erdepa ASC,erage DESC;
```

3.4 选　　择

选择操作从关系表中选择一些满足特定条件的元组行，比如选择属性满足一些条件（谓词表达式）的元组行，或者无条件选择关系表中的所有元组行。

[例 3-12] 查询 event 表中所有事件本身的属性信息。

```
SELECT *
FROM event;
```

[例 3-13] 查询 event 表中事件地点是北京的所有信息。

```
SELECT *
FROM event
WHERE event_place ='北京';
```

通常，保留字如 FROM、WHERE 等另起一行与 SELECT 左对齐，这种风格显得直观易读；FROM 子句给出查询所涉及的关系表；WHERE 子句给出查询条件，像关系代数中的选择条件，只有满足这个条件的元组行才出现在查询结果中；SELECT 子句给出满足查询条件元组行的哪些属性（或目标列表达式）出现在查询结果中。SELECT-FROM-WHERE 查询的一种解释是，首先考虑 FROM 子句提及关系表中的每个元组行，选择出其中使 WHERE 子句条件为真的，按 SELECT 子句出现的属性序列构造查询结果中的

元组行。WHERE 子句中的条件表达式可以用各种运算符组合而成，常用的运算符如表 3-17 所示。

表 3-17　常用的运算符

运算符名称	符号及格式	说　明
比较	=,>,<,>=,<=,!=,<>,!>,!<	比较两个表达式的值
确定范围	<表达式 1>[NOT] BETWEEN <表达式 2> AND<表达式 3>	(不)在给定范围上
是否空值	<表达式 1> IS [NOT] NULL	判断是否为空
是否属于集合	<元组行> [NOT]IN <集合>	判断某元组行是否在某集合内
限定比较判断	<元组行> ALL|SOME|ANY(<集合>)	元组行与集合中每一个或某一个元组行满足比较判断限定
存在判断	[NOT]EXISTS(<集合>)	判断集合是否存在一个元组行
唯一判断	[NOT]UNIQUE(<集合>)	判断集合是否没有重复元组行
串匹配	[NOT]LIKE	%：与零个或多个字符组成的字符串匹配；_：与单个字符匹配；ESCAP：定义转义符
逻辑表达式	AND,OR,NOT	复合多个查询条件

下面给出一些典型选择查询的示例，其中有些也涉及投影。

1. 比较选择

[例 3-14]　查询历史学院全体考官。

```
SELECT *
FROM examiner
WHERE erdepa='历史学院';
```

SQL Server 中使用单引号作字符串常量的标识，对于包含单引号的字符串，使用两个单引号表示一个单引号，注意不能写成双引号。

[例 3-15]　查询所有工资在 5800 元以上的考官，按工资数升序排列。

```
SELECT *
FROM examiner
WHERE ersalary>5800
ORDER BY ersalary;
```

[例 3-16]　查询工资不到 10 000 元的考官。

```
SELECT *
FROM examiner
WHERE ersalary<=10000;
```

2. 范围选择

［例 3-17］ 查询工资为 6000～9000 元(包括 6000 元和 9000 元)的考官,按工资降序排列。

```
SELECT *
FROM examiner
WHERE ersalary BETWEEN 6000 AND 9000
ORDER BY ersalary DESC;
```

［例 3-18］ 查询工资不为 16 000～19 000 元(包括 16 000 元和 19 000 元)的考官。

```
SELECT *
FROM examiner
WHERE ersalary NOT BETWEEN 16000 AND 19000;
```

3. 空值选择

［例 3-19］ 查询尚未登记联系电话的院系。

```
SELECT *
FROM department
WHERE dtele IS NULL;
```

［例 3-20］ 查询已经登记联系电话的院系。

```
SELECT *
FROM department
WHERE dtele IS NOT NULL;
```

4. 集合归属选择

［例 3-21］ 查询历史学院和心理学院考官。

```
SELECT *
FROM examiner
WHERE erdepa IN ('历史学院', '心理学院');
```

［例 3-22］ 查询既不是历史学院也不是心理学院的考官。

```
SELECT *
FROM examiner
WHERE erdepa NOT IN ('历史学院', '心理学院');
```

5. 串匹配选择

字符串的匹配可以使用“=”“LIKE”及正则表达式运算符。“=”要求两边字符串严格相同;“LIKE”允许使用通配符“_”(下画线匹配任何单个字符)和“%”(一个百分号匹配零个或多个任何字符),如果不使用通配符,“LIKE”和“=”等价。正则表达式中,“_” 表示任意一个字符;“*”表示前面的字符出现任意多次或 0 次;“+”表示前面的字符至少出现一次;“[]”表示一个字符集合中任意一个字符。正则表达式运算符和 LIKE 一样,默认转义符为“\”,也可以用 ESCAPE 定义别的字符作为转义符。转义符后的元字符及所定义的转义符都作为普通字符本身。

1) 匹配串为固定字符串

[例 3-23]　查询联系电话为 58807998 的院系。

```
SELECT *
FROM department
WHERE dtele LIKE '58807998';
```

等价于:

```
SELECT *
FROM department
WHERE dtele='58807998';
```

2) 匹配串为含通配符的字符串

[例 3-24]　查询所有以学院命名的院系,如文学院、数学科学学院等。

```
SELECT *
FROM department
WHERE dname LIKE '%学院';
```

[例 3-25]　查询所有以系命名,且全名为三个汉字的院系,如物理系、天文系等。

```
SELECT *
FROM department
WHERE dname LIKE '___系';
```

当数据库字符集是 ASCII 时,每个汉字占位两个字符,通配符相当于两个下画线。

[例 3-26]　查询院系名中第 2 个字为“学”字的院系,如文学院、化学系等。

```
SELECT *
FROM department
WHERE dname LIKE '_学%';
```

[例 3-27]　查询所有院系名不以“教育”开头的院系。

```
SELECT *
FROM department
WHERE dname NOT LIKE '教育%';
```

等价于：

```
SELECT *
FROM department
WHERE charindex('教育',dname)<=0;
```

3）使用转义字符将通配符转义为普通字符

[例 3-28] 查询大学外语试卷的试卷号和类别。

```
SELECT eid,etype
FROM exampaper
WHERE ename LIKE '大学外语';
```

[例 3-29] 查询以“数据库_”开头的试卷。

```
SELECT *
FROM exampaper
WHERE ename LIKE '数据库*_%' ESCAPE '*';
```

其中，ESCAPE '*' 表示此处定义“*”为转义字符。

6. 逻辑表达式选择

[例 3-30] 查询历史学院年龄在 58 岁以上、工资尚不足 15 000 元的考官。

```
SELECT *
FROM examiner
WHERE erdepa='历史学院' AND erage>58 AND ersalary<=15000;
```

[例 3-31] 查询历史学院和心理学院的女考官。

```
SELECT *
FROM examiner
WHERE ersex='女' AND (erdepa='历史学院' OR erdepa='心理学院');
```

基于 ALL、SOME、ANY 的限定比较选择，基于 EXISTS、NOT EXISTS 的存在性选择和基于 UNIQUE、NOT UNIQUE 的唯一性选择，大多出现在 3.10 节所述的嵌套查询中。

3.5　集合操作

并、交、差查询对应于数学集合论中的∪、∩和-运算。当两个子查询结果的结构完全一致时，可以让这两个子查询执行并、交、差操作。并、交、差的运算符为 UNION、INTERSECT 和 EXCEPT。

```
(SELECT 查询语句 1)
UNION [ALL]
(SELECT 查询语句 2)


(SELECT 查询语句 1)
INTERSECT [ALL]
(SELECT 查询语句 2)


(SELECT 查询语句 1)
EXCEPT [ALL]
(SELECT 查询语句 2)
```

为了能够计算两个查询的并、交、差，这两个查询必须是“兼容的”，也就是它们都有同样数量的列，并且对应列的数据类型是兼容的，集合操作中不带保留字 ALL 时，默认像 DISTINCT 那样删除结果中所有重复的行，返回结果表自动消除重复元组；而声明了 ALL 时，返回结果保留重复元组。集合操作也可以嵌套和级联。

如果表 t_1 中行 r 出现 n 次，而 t_2 中行 r 出现 m 次，那么在 TABLE t_1 UNION ALL TABLE t_2 中行 r 将出现 $n+m$ 次；在 TABLE t_1 EXCEPT ALL TABLE t_2 中，行 r 将出现 $\max(0, n-m)$ 次；在 TABLE t_1 INTERSECT ALL TABLE t_2 中，行 r 将出现 $\min(n,m)$ 次。其中，n 或 m（或两者）可以为 0。

[例 3-32]　查询历史学院的考官和工资不到 6000 元的考官。

方法一：

```
SELECT *
FROM examiner
WHERE erdepa='历史学院'
UNION
SELECT *
FROM examiner
WHERE ersalary<=6000;
```

方法二：

```
SELECT DISTINCT *
FROM examiner
WHERE erdepa='历史学院' OR ersalary<=6000;
```

[例 3-33] 查询历史学院年龄不到 50 岁的考官。

```
SELECT *
FROM examiner
WHERE erdepa='历史学院'
EXCEPT
SELECT *
FROM examiner
WHERE erage>=50;
```

3.6 连接查询

可以通过内连接(inner join)、外连接(outer join)左连接(left join)、右连接(right join)、全连接(full join)、交叉连接(cross join)等实现多表查询,即从多个表中进行查询。

对于两个关系表的连接操作,连接操作符分成连接类型和连接条件两部分。连接类型决定了如何处理连接条件中不匹配的元组,即分为内连接和(左/右/全)外连接。连接条件决定了两个关系表中哪些元组应该匹配,以及连接结果中出现哪些属性,其中,如果条件为永真则等价于交叉连接;如果条件为全部共同属性值相等则等价于自然连接。

SQL Server 中连接的一般形式如下。

```
t1 { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } JOIN t2 ON <逻辑表达式>
```

INNER 和 OUTER 对所有连接类型都是可选的。默认为 INNER;LEFT、RIGHT、FULL 均隐含外连接。连接条件在 ON 子句中声明,连接条件用来判断来自两个源表中的哪些行是“匹配”的。

ON 子句接收一个和 WHERE 子句相同的布尔表达式,如果两个分别来自 R_1 和 R_2 的元组在 ON 表达式上运算的结果为真,那么它们就算是匹配的行。

3.6.1 笛卡儿积(交叉连接)

基本查询包括三个子句:SELECT 子句、FROM 子句和 WHERE 子句。在写多表连接查询语句时,查询涉及多个表,最简单直接的方法就是在 FROM 后面依次写上这些表名,并以逗号或 CROSS JOIN 分隔,FROM 子句的结果表就是这些表的笛卡儿积,结果表包含所有这些表的所有列,如果两个表中有同名列,在列名前加上表名作前缀,表明该列的来源表。根据查询需要,还可通过 WHERE 子句对笛卡儿积结果表施加选择操作,以撷取那些符合查询条件的行。如果不带 WHERE 条件子句,它将会返回被连接的两个表的笛卡儿积,返回结果的行数等于两个表行数的乘积,这种交叉连接的结果无实际意

义；如果带 WHERE 条件子句，返回或显示的是匹配的行数。

［例 3-34］　查询每个考生及其报考试卷的情况。

```
SELECT *
FROM examinee,eeexam /* 等价于 FROM examinee CROSS JOIN eeexam */
WHERE examinee.eeid =eeexam.eeid;
```

通过笛卡儿积把来自 examinee 和 eeexam 中具有相同 eeid 的元组行进行匹配。

［例 3-35］　各个单位参与预案编制的情况。

```
SELECT *
FROM departments, record, plan
WHERE departments.depart_id= record.depart_id AND record.plan_id=plan.plan
_id;
```

［例 3-36］　查询报考 0205000002 试卷且成绩在 90 分以上的所有考生的考试号和姓名。

```
SELECT examinee.eeid, eename
FROM examinee, eeexam
WHERE examinee.eeid=eeexam.eeid AND
      eeexam.eid='0205000002'AND
      eeexam.achieve >90;
```

3.6.2　内连接

条件内连接，是在笛卡儿积运算的基础上选取满足给定条件的行。内连接有等值连接、不等值连接和自然连接这三种。条件连接把一个选择运算和一个笛卡儿积运算合并为单独的一个运算，用 JOIN…ON…实现。当连接操作符是“＝”时，该连接操作被称为等值连接，使用其他运算符的连接运算符称为不等值连接。当等值连接中的连接字段相同且在 SELECT 语句中的输出列表中去除了重复字段时，该连接操作为自然连接。

［例 3-37］　检索考官姓名及所在学院办公电话。

```
SELECT ername, dtele
FROM examiner JOIN department
ON examiner.erdepa=department.dname;
```

或：

```
SELECT ername, dtele
FROM examiner INNER JOIN department
ON examiner.erdepa=department.dname;
```

［例 3-38］　参与 1 号预案编制的单位名称和电话。

```
SELECT depart_name, depart_tel
FROM departments JOIN record
ON departments.depart_id=record.depart_id where plan_id=1;
```

3.6.3 外连接

内连接可能会出现左表当中的一些行在右表中没有相匹配的行，或右表当中的一些行在左表中没有相匹配的行，这些没有找到匹配的行称为悬浮行。

内连接和外连接的区别就在于对悬浮行的处理不同。

内连接抛弃所有悬浮行。外连接运算对悬浮行的处理有三种方式：左外连接，右外连接，全外连接，三种方式都要首先计算内连接，然后再在内连接的结果中加上相应的左(右、左和右)表中的悬浮行。

例如，左外连接是这样计算的：首先，计算内连接的结果；然后，把左侧表中的悬浮行加入结果表，这些行中来自右侧表的属性赋为空值 null。

与左外连接类似，右外连接就是把右侧表中的悬浮行补上空值后加入结果表。

全外连接是左外连接和右外连接的组合，即两个表中所有的行都会出现在结果集中，左侧表中的悬浮行补上空值后加到结果表中，同时，右侧表中的悬浮行补上空值后也加到结果表中。

外连接运算分别用 LEFT OUTER JOIN…ON…、RIGHT OUTER JOIN…ON…FULL OUTER JOIN…ON…等来实现，使用中可以省略 OUTER。

[例 3-39] 关系 examiner 和 department 基于学院名相等的左外连接。

```
SELECT ername, dtele
FROM examiner LEFT OUTER JOIN department
ON examiner.erdepa=department.dname;
```

或：

```
SELECT ername, dtele
FROM examiner LEFT JOIN department
ON examiner.erdepa=department.dname;
```

3.7 更　　名

有时，一个表在 FROM 子句中多次出现，即这个表被多次引用。为区别不同的引用，应给表或表引用起一个临时的表别名，语法如下。

```
FROM 表 [AS] 别名
```

取了别名之后就不允许再用最初的名字了。

[例 3-40] 查询同院系工作的两位考官。