数据库： SQL - 第2部分

课程和练习

数据操作语言

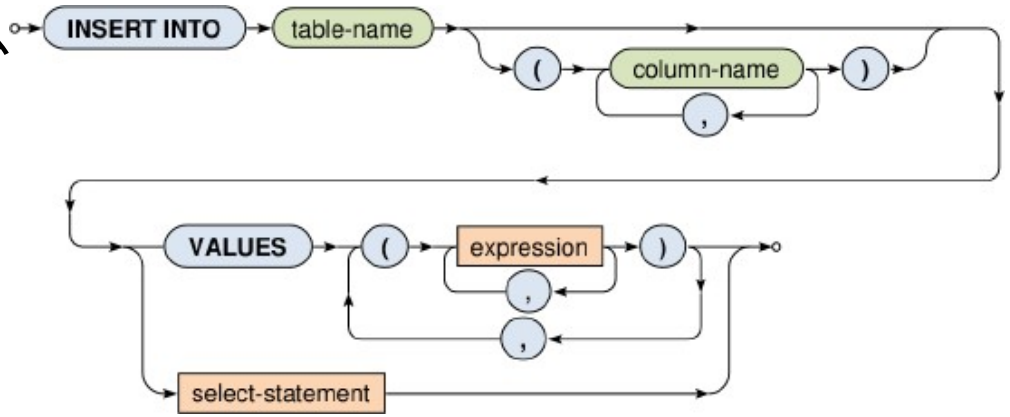
插入、更新、删除选择

数据修改语言

- **INSERT INTO** - 插入行数
- **UPDATE** - 改变行数
- **DELETE FROM** - 删除记录

插入

- 在表中插入新行
  - 用明确的枚举法
  - 从一个SELECT的结果来看
- 在缺少列的情况下，将采用默认值
- 如果没有默认值，那么



INSERT INTO: 例子

```
CREATE TABLE Product (
    id INTEGER PRIMARY KEY,
    name VARCHAR(128) UNIQUE、
    price DECIMAL(6,2) NOT NULL、
    生产日期、
    available BOOLEAN DEFAULT TRUE,
    weight FLOAT、
    生产者 INTEGER
);
```

```
INSERT INTO Product VALUES (0, 'Chair1', 2000, '2015-05-06', TRUE, 3.5, 11);
```

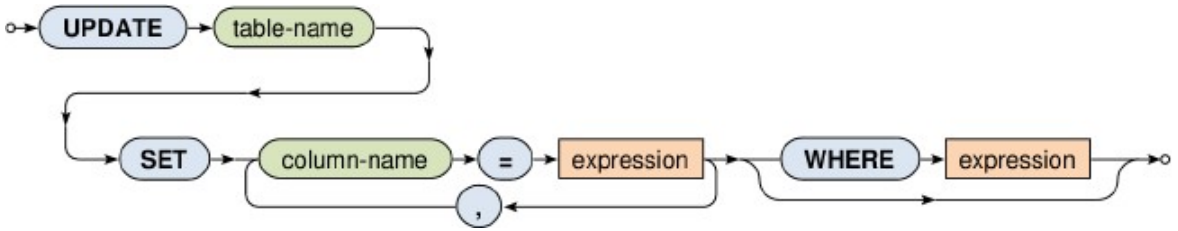
没有提到**可用的**列，插入的行将采用默认值（TRUE）。

插入产品

```
(id, name, price, produced, weight, producer)
VALUES (1, 'Chair2', 1500, '2015-05-06', 4.5, 11) ;
```

UPDATE WHERE

- 编辑表格中的现有行
  - 只有符合条件的线
- 新的分配值可以是：
  - NULL，字面意思，由表达式给出的值，子查询的结果



UPDATE WHERE: 示例

```
CREATE TABLE Product (
    id INTEGER PRIMARY KEY,
    name VARCHAR(128) UNIQUE、
    price DECIMAL(6,2) NOT NULL、
    生产日期、
    available BOOLEAN DEFAULT TRUE,
    weight FLOAT、
    生产者 INTEGER
);
```

设置名称='笔记本

WHERE (name = 'Laptop');

- *将"笔记本电脑"改为"笔记本*

  UPDATE 产品

- *更新价格，对2015年1月1日前生产的产品给予10%的折扣*

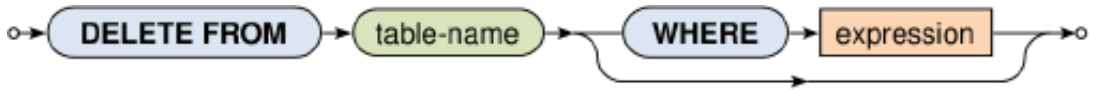
UPDATE 产品

设置价格 = 价格 * 0.9

WHERE (Produced < '2015-01-01');

DELETE FROM

- 从表中删除现有行
  - 只有符合条件的线



DELETE FROM: 例子

```
CREATE TABLE Product (
    id INTEGER PRIMARY KEY,
    name VARCHAR(128) UNIQUE、
    price DECIMAL(6,2) NOT NULL、
    生产日期、
    available BOOLEAN DEFAULT TRUE,
    weight FLOAT、
    生产者 INTEGER
);
```

删除所有价格高于1000的产品

```
DELETE FROM Product

WHERE (Price > 1000);
```

选择

- 允许你在.NET中查询一个数据库：
  - 选择表格中的某些列：**投影**
  - 根据内容选择表格中的某些行：**选择**
  - 合并几个表的信息：**连接、联合、交叉、差异和分割**
  - 结合这些不同的操作
- 查询是对表的操作的组合，其结果本身是一个表，其存在是短暂的（查询的时间）。

挑选：投影

- 语法：
  - **SELECT** column1 **FROM** tableA ➔ 检索一个特定的列
  - **SELECT * FROM** tableA 或 **SELECT** tableA.* **FROM** tableA ➔ 检索所有列
  - **SELECT DISTINCT** column1 **FROM** tableA ➔ 检索一个没有重复的特定列
    - **ALL** (默认) ➔ 所有的行都出现在结果中。
  - **SELECT** column1 **AS** C1, column2 **AS** C2 **FROM** tableA **AS** TA ➔ 别名

- **SELECT concat**(column1, ' ',column_2) AS column FROM T_CLIENT ➔ 串联列

SELECT: 投影的例子

Flights:

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
LH438	Lufthansa	Stuttgart	68
OK012	CSA	Milano	37
OK321	CSA	London	156
AC906	Air Canada	Toronto	116
KL7621	KLM	Rotterdam	75
KL1245	KLM	Amsterdam	130

Aircrafts:

Aircraft	Company	Capacity
Boeing 717	CSA	106
Airbus A380	KLM	555
Airbus A350	KLM	253

- **SELECT ALL** * FROM Aircrafts

Aircraft	Company	Capacity
Boeing 717	CSA	106
Airbus A380	KLM	555
Airbus A350	KLM	253

- **SELECT** Company FROM Aircrafts

Company
CSA
KLM
KLM

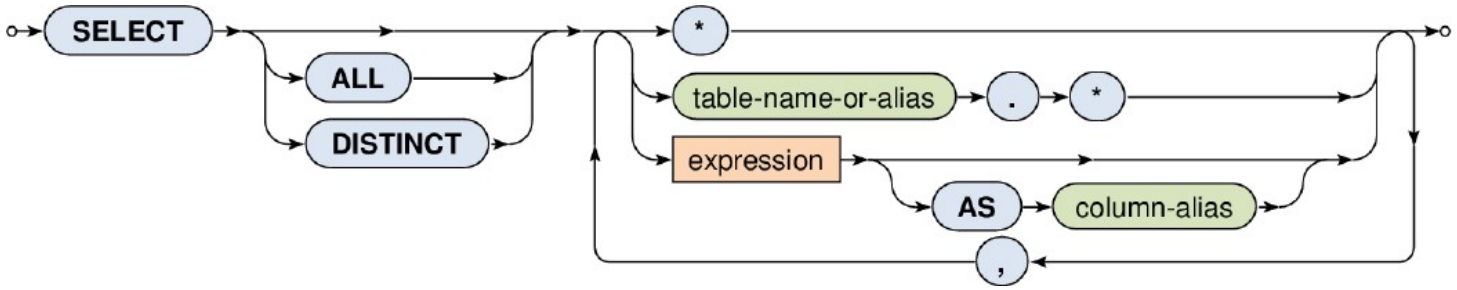
- **SELECT DISTINCT** Company FROM Aircrafts

Company
CSA
KLM

- **SELECT DISTINCT** Company **AS** Carrier FROM Aircrafts

运营商
CSA
KLM

SELECT： 投影(例子)



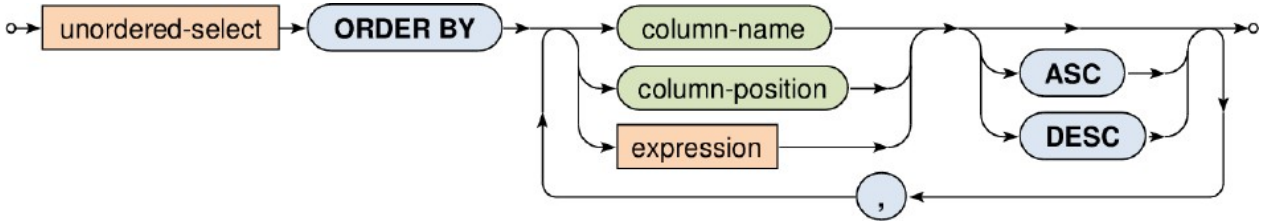
选择：按顺序

- 允许你对列进行排序
  - 或者通过指定列的字面名称
  - 或者通过在SELECT关键字后面的枚举中指定其顺序号
- ASC（默认）或DESC
- **选择**列1，列2

FROM表A

ORDER BY column2

选择列1，列2
FROM表A
ORDER BY 2



SELECT : ORDER BY (例子)

Flights:

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
LH438	Lufthansa	Stuttgart	68
OK012	CSA	Milano	37
OK321	CSA	London	156
AC906	Air Canada	Toronto	116
KL7621	KLM	Rotterdam	75
KL1245	KLM	Amsterdam	130

Aircrafts:

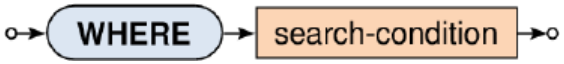
Aircraft	Company	Capacity
Boeing 717	CSA	106
Airbus A380	KLM	555
Airbus A350	KLM	253

- 返回所有已编程的目的地的有序列表

```
SELECT DISTINCT Destination
发出航班
ORDER BY Destination ASC
```

Destination
Amsterdam
London
Milano
New York
Rotterdam
Stuttgart
Toronto

选择：WHERE选择



- 代表一个行必须满足的选择条件，以便出现在结果中。
- 使用可通过以下方式组合的表达式
  和、OR和NOT
- 例子：
  - ... **凡**（容量>200）**和**（飞机**LIKE**'Airbus%'） ...
  - ... **WHERE** (Company **IN** ('KLM', 'Emirates')) ...
  - ……**如果不是**（乘客人数**在**100和200**之间**）……

SELECT : WHERE (例子)

Flights:

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
LH438	Lufthansa	Stuttgart	68
OK012	CSA	Milano	37
OK321	CSA	London	156
AC906	Air Canada	Toronto	116
KL7621	KLM	Rotterdam	75
KL1245	KLM	Amsterdam	130

Aircrafts:

Aircraft	Company	Capacity
Boeing 717	CSA	106
Airbus A380	KLM	555
Airbus A350	KLM	253

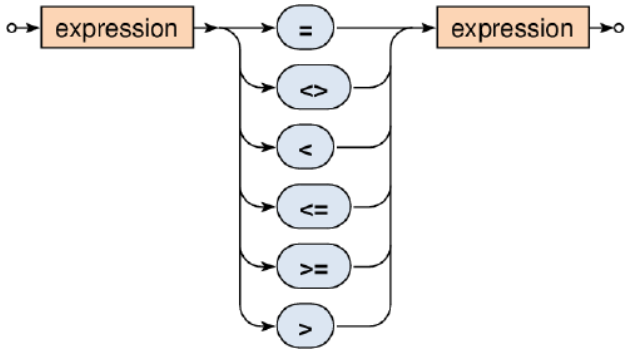
- 选择 *
  发出航班
  `WHERE Company='KLM'`

Flight	Company	Destination	Passengers
KL7621	KLM	Rotterdam	75
KL1245	KLM	Amsterdam	130

-
```
SELECT Destination, Passengers
FROM Flights
WHERE Company='KLM' AND Passengers>100
```

Destination	Passengers
Amsterdam	130

SELECT : WHERE和操作者 (1)

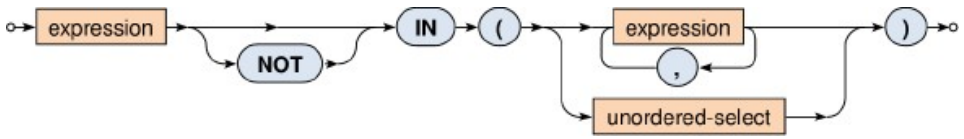


- **比较**运算符

- **BETWEEN** 区间运算符
  - **value BETWEEN Min AND Max相当于(Min <= value) AND (value <= Max)**
  - **例：……乘客人数在**100和200**之间的情况下**

SELECT : WHERE和操作者 (2)

- **IN**运算符： 检查一个数据集中是否存在一个值（或不存在）。
  - 例： 公司 **IN** ('KLM', 'Emirates')
  - 例： 公司**不在**('KLM', 'Emirates')



SELECT : WHERE和操作者 (3)

- **LIKE**操作符
  - 允许进行部分比较
  - 多用于包含α数据的列
  - 使用通配符%和_
    - 替换任何字符串，包括空字符串。
    - _ 替换一个且仅有一个字符
- 例子：
  - LIKE 'B%' : 以B开头的值
  - LIKE '%B' : 值以B结尾
  - LIKE '_B%': 在第二个位置包含'B'的值
  - LIKE 'B__%'： 以B开头且至少有3个字符的值

- LIKE 'A%B'：以A开头， 以B结尾的值

SELECT: 聚合功能

- **AVG()**计算一组数值的平均值。
  - 例如：每个客户的平均篮子价格
- **COUNT()**对涉及的行数进行统计。
  - 例如：每个客户购买了多少东西
- **MAX()**检索的是最高值
  - 例如：最昂贵的产品
- **MIN()**检索最小的值。
  - 例如：客户第一次购买的日期
- **SUM()**计算几行的总和
  - 例如：一个客户的所有购买行为的总和

SELECT: 聚合功能(示例)

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
LH438	Lufthansa	Stuttgart	68
OK012	CSA	Milano	37
OK321	CSA	London	156
AC906	Air Canada	Toronto	116
KL7621	KLM	Rotterdam	75
KL1245	KLM	Amsterdam	130

选择

COUNT(*) **AS** Flights、

COUNT(**DISTINCT** Company) **AS** Companies、

累计(乘客) **硕** PSum

士、

平均数(乘客) **硕** ĀĀĀ

士

闵行区(乘客) **硕** 硕士生

士、

MAX（乘客） **硕士** 医学博士

发出航班

Flights	Companies	PSum	PAvg	PMin	PMax
7	4	858	123	37	276

选择：按分组

- 将在摘要中具有相同值的行进行分组
  - 例：找出每个国家的客户数量

SELECT: HAVING

- 与WHERE类似，只是它允许使用聚合函数进行过滤
- HAVING经常与GROUP BY一起使用，但它不是强制性的。

SELECT : HAVING (例子)

- 每个公司计划了多少次飞行?
  - 然而，我们对前往斯图加特和慕尼黑的航班不感兴趣
  - 而且我们不希望那些飞过一次或更少的公司

SELECT : HAVING (例子)

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
LH438	Lufthansa	Stuttgart	68
OK012	CSA	Milano	37
OK321	CSA	London	156
AC906	Air Canada	Toronto	116
KL7621	KLM	Rotterdam	75
KL1245	KLM	Amsterdam	130

⇨

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
OK012		Milano	37
OK321		London	156
AC906	Air Canada	Toronto	116
KL7621	KLM	Rotterdam	75
KL1245		Amsterdam	130

⇨

Company	Flights
CSA	3
Air Canada	1
KLM	2

⇩

Company	Flights
CSA	3
KLM	2

```
SELECT 公司, COUNT(*) AS Flights
```

发出航班

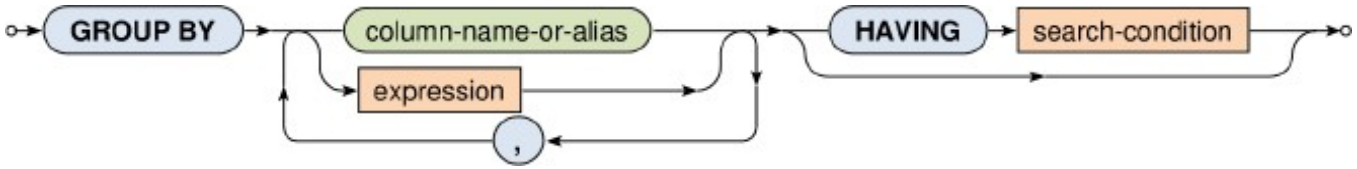
```
WHERE (Destination NOT IN ('Stuttgart', 'Munich'))
GROUP BY Company HAVING (Flights > 1)
```

isep
École d'ingénieurs du numérique

SELECT: 订单

SELECT *column_name(s)*
FROM *table_name*
WHERE *condition*
GROUP BY *column_name(s)*
具备条件

- 首先...
  - 对**FROM**和**WHERE**子句进行评估
  - 这在于一个中间的表格中
- 然后...
  - 这个中间表的列根据**GROUP BY**中指定的相同的列被划分为组。
- 最后
  - 这些汇总的列可以使用**HAVING**条件进行过滤



从

- **定义了要查询的表**
- 有两种方法可以做到这一点：
  - **记号 WHERE**
    - 以逗号分隔的表格列表
    - 假设它们的行的**笛卡尔乘积**为
    - 连接条件是在WHERE子句中指定的

```
WHERE Table1.xxx =
  Table2.yyy AND condition
```

选择...

```
FROM 表1, 表2
```

- **带有不同JOIN操作符的符号**

选择...

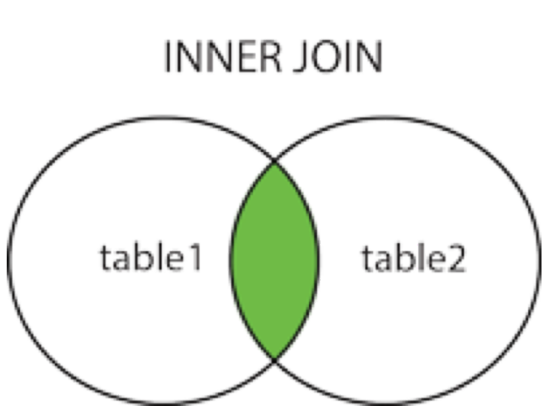
```
FROM Table1 JOIN Table2 WHERE
condition
```

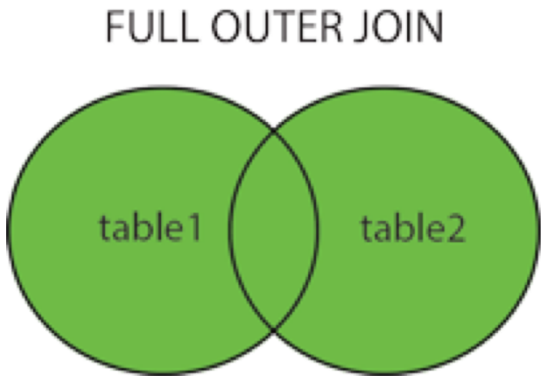
来自: 加入

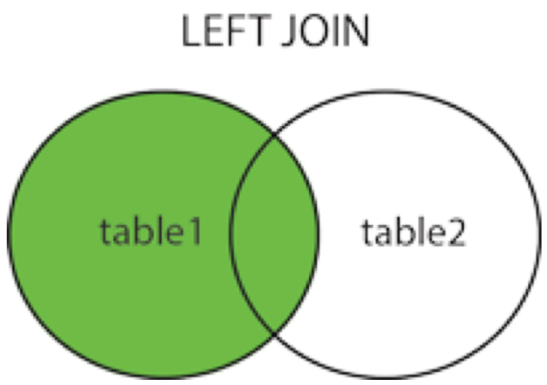
- CROSS JOIN或JOIN
- 自然连接
- INNER JOIN
- CROSS JOIN
- LEFT JOIN（或LEFT OUTER JOIN）。
- RIGHT JOIN（或RIGHT OUTER JOIN）。
- 全连接（或全外连接）。
- UNION JOIN

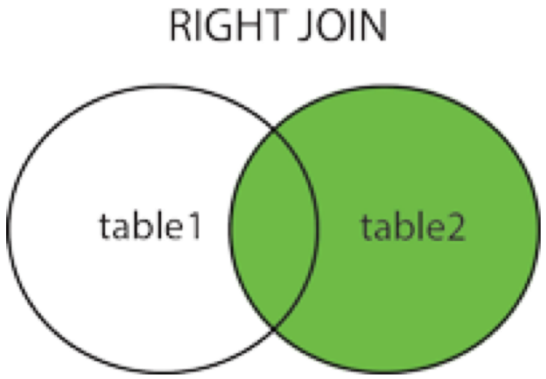
isep
École d'ingénieurs du numérique

来自: 加入


INNER JOIN
table1
table2

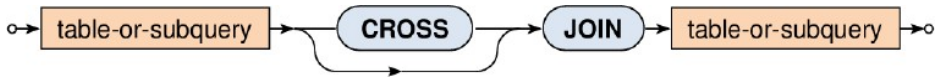

FULL OUTER JOIN
table1
table2


LEFT JOIN
table1
table2


RIGHT JOIN
table1
table2

W3SCHOOL来源

来自: (Cross) 加入



- 交叉连接允许对2个表进行笛卡尔乘积，即把一个表的每一行与第二个表的每一行连接起来。请注意，结果的数量通常非常多。
  - 选择 *从 T1 **横向** **加入** T2
  - 选择 *从 T1 **加入** T2

记号 其中：
选择 * 来自
+1 +2

A	T1.*
1	...
2	...
3	...

A	T2.*
1	...
4	...

T1.A	T1.*	T2.A	T2.*
1	...	1	...
1	...	4	...
2	...	1	...
2	...	4	...
3	...	1	...
3	...	4	...

来自：自然加入

- 如果2个SQL表之间至少有**一个同名的列，则**在2个表之间进行自然连接
  - 即同名的栏目

记号 其中：
选择 * 来自
t1, t2

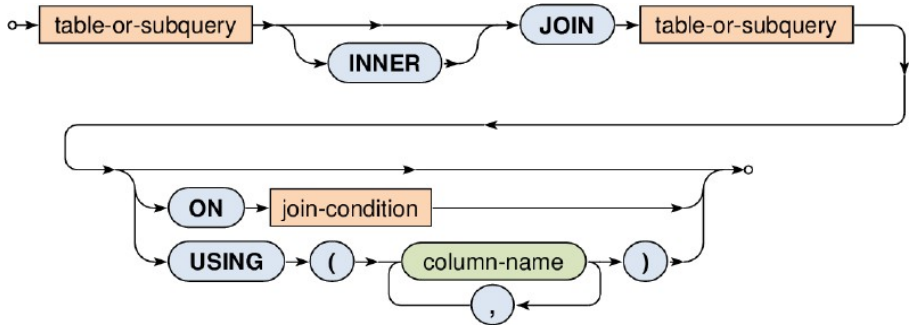
- `select * from t1 natural join t2`

A	T1.*
1	...
2	...
3	...

A	T2.*
1	...
4	...

A	T1.*	T2.*
1	...	...

FROM: 内联



- 内部连接，当**ON**关键字后面的条件为真时返回记录。
  - 这是最常见的关节之一。
  - `select * from t1 inner join t2 on (t1.a <= t2.a)`
  - `select * from t1 join t2 on (t1.a <= t2.a)`

A	T1.*
1	…
2	…
3	…

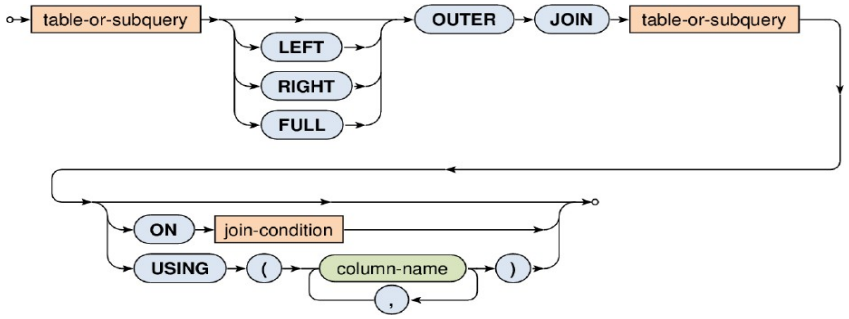
A	T2.*
1	…
4	…

T1.A	T1.*	T2.A	T2.*
1	…	1	…
1	…	4	…
2	…	4	…
3	…	4	…

凡是记号：

```
SELECT *
FROM T1, T2
```

FROM: 外联



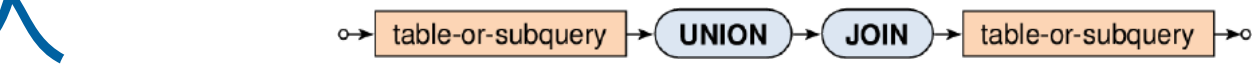
- 外部连接返回左表（**LEFT OUTER JOIN**）或右表（**RIGHT OUTER JOIN**）的所有记录，即使条件在另一个表中不成立。
  - **完整**（默认）：当条件在2个表中至少有一个为真时。
  - `select * from t1 left outer join t2 on (t1.a = t2.a)`

A	T1.*
1	...
2	...
3	...

A	T2.*
1	...
4	...

T1.A	T1.*	T2.A	T2.*
1	...	1	...
2	...	NULL	NULL
3	...	NULL	NULL

发件人：联盟加入



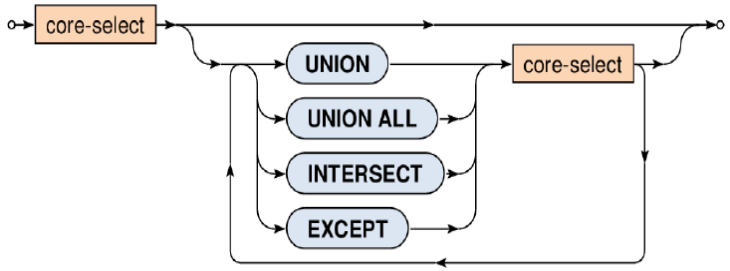
- 两个表的行都整合到一个表中，没有行的组合
  - `select * from t1` **`union join`** `t2`

A	T1.*
1	…
2	…
3	…

A	T2.*
1	…
4	…

T1.A	T1.*	T2.A	T2.*
1	…	NULL	NULL
2	…	NULL	NULL
3	…	NULL	NULL
NULL	NULL	1	…
NULL	NULL	4	…

设置运算符



- **UNION**：两个表的联合，没有重复。
- **UNION ALL** : 两个有重复的表的联合
- **INTERSECT**: 两个表格的交叉点
- **除外**：两个表格之间的差异

- **两个操作数必须是兼容的**
  - 相同的列数

- 这些列必须是相同的类型

集合运算符：例子

Flights:

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
LH438	Lufthansa	Stuttgart	68
OK012	CSA	Milano	37
OK321	CSA	London	156
AC906	Air Canada	Toronto	116
KL7621	KLM	Rotterdam	75
KL1245	KLM	Amsterdam	130

Aircrafts:

Aircraft	Company	Capacity
Boeing 717	CSA	106
Airbus A380	KLM	555
Airbus A350	KLM	253

```
SELECT Company FROM Flights
```

联 盟

```
SELECT Company
FROM Aircrafts
```

Company
CSA
Lufthansa
Air Canada
KLM

```
SELECT Company FROM
Flights
INTERSECT
SEL    ny
FRO    ts
```

Company
CSA
KLM

```
SELECT Company FROM Flights
```

除外

```
SELECT Company
FROM Aircrafts
```

Company
Lufthansa
Air Canada

嵌套查询

- 一个请求在另一个请求中，它可以被用来：
  - 在一个操作员之后
    - **任何、某些、所有**
    - **纳入**
    - **存在的**
  - 当在FROM子句中定义一个表时
  - 如果产生标量值，几乎所有的表达式

嵌套查询：实例1

- 查找所有乘客人数高于平均水平的定期航班

Flights:

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
LH438	Lufthansa	Stuttgart	68
OK012	CSA	Milano	37
OK321	CSA	London	156
AC906	Air Canada	Toronto	116
KL7621	KLM	Rotterdam	75
KL1245	KLM	Amsterdam	130

Aircrafts:

Aircraft	Company	Capacity
Boeing 717	CSA	106
Airbus A380	KLM	555
Airbus A350	KLM	253

选择 *

发出航班

```
WHERE (Passengers > (SELECT AVG(Passengers) FROM Flights))
```

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
OK321	CSA	London	156
KL1245	KLM	Amsterdam	130

嵌套查询：实例2

- 返回每个航班的适当数量的飞机。
  - 只有特定公司的、有足够能力的飞机可以使用
  - 请注意外部查询的值是如何与内部查询相关的

Flights:

Flight	Company	Destination	Passengers
OK251	CSA	New York	276
LH438	Lufthansa	Stuttgart	68
OK012	CSA	Milano	37
OK321	CSA	London	156
AC906	Air Canada	Toronto	116
KL7621	KLM	Rotterdam	75
KL1245	KLM	Amsterdam	130

Aircrafts:

Aircraft	Company	Capacity
Boeing 717	CSA	106
Airbus A380	KLM	555
Airbus A350	KLM	253

Flight	Company	Destination	Passengers	Aircrafts
OK251	CSA	New York	276	0
LH438	Lufthansa	Stuttgart	68	0
OK012	CSA	Milano	37	1
OK321	CSA	London	156	0
AC906	Air Canada	Toronto	116	0
KL7621	KLM	Rotterdam	75	2
KL1245	KLM	Amsterdam	130	2

选择

```
航线.*, (
    选择 count(*)
    From Aircrafts AS A
    WHERE
        (A.公司=F.公司) AND (A.容量>=F.
        乘客)
) AS飞机
```

从航班中选出F

存储系统：MyISAM vs InnoDB

我的ISAM

+

- 默认系统MySQL
- 快速SELECT或INSERT+查询

-

- 没有交易
- 没有外键
- 崩溃后更难恢复

基础设施

+

- 管理交易
- 处理外键和完整性约束
- 支持ACID，以确保所有注册成功或失败。

-

- 较大的存储引擎。它需要更多的资源，而且速度较慢