|  |  |
| --- | --- |
| （标题）**Querying Data**  SELECT 的使用 | |
| （概要）   1. 通常SELECT与FROM配合使用，可以指定SELECT的字段 2. “\*”代表所有字段 3. SELECT多个字段 4. SELECT等关键字通常使用大写，如果表名或者字段名中包含大写，则需要将通过双引号包起来 5. SELECT拼接内容 6. 给拼接内容赋名 7. 计算数值 8. 查看pg版本 | （详情） |
| SELECT first\_name FROM customer; |
| SELECT \* FROM customer; |
| SELECT first\_name, last\_name FROM customer; |
| SELECT “FIRST\_NAME” FROM customer; |
| SELECT first\_name || ‘ ’ || last\_name FROM customer; |
| SELECT first\_name || ‘ ‘ || last\_name AS full\_name FROM customer; |
| SELECT 5 \* 3 AS result |
| SELECT version(); |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结）   1. SELECT与FROM配合使用 2. SELECT的对象如果复杂，可以赋“别名” | |

|  |  |
| --- | --- |
| （标题）**Querying Data**  ORDER BY的使用 | |
| （概要）   1. ORDER BY 默认是升序 2. ASC：升序 3. DESC：降序 4. 多个排序是，排序对象逗号分隔 5. 通过expression排序 | （详情） |
| SELECT first\_name FROM customer ORDER BY first\_name |
| SELECT first\_name FROM customer ORDER BY first\_name ASC |
| SELECT first\_name FROM customer ORDER BY first\_name DESC |
| SELECT first\_name, last\_name FROM customer ORDER BY first\_name ASC, last\_name DESC |
| SELECT first\_name LENGTH（first\_name） len FROM customer ORDER BY first\_name |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结）  ORDER BY配合SELECT一起使用，ORDER BY的对象是字段，方法有“升序”和“降序” | |

|  |  |
| --- | --- |
| （标题）**Querying Data**  SELECT DISTINCT的使用 | |
| （概要）   1. 指定一个字段 2. 指定两个字段，联合唯一 3. 指定一个字段并SELECT关联字段 4. 指定两个字段并SELECT关联字段 5. 与ORDER BY配合使用 | （详情）去除重复数据，保证被SELECT字段唯一性 |
| SELECT DISTINCT first\_name FROM customer; |
| SELECT DISTINCT length, rating FROM film; |
| SELECT DISTINCT ON (rating) title, rating, length FROM film; |
| SELECT DISTINCT ON (rating, length) title, rating, length FROM film; |
| SELECT DISTINCT ON (rating) length, rating FROM film ORDER BY rating; |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结）  1、DISTINCT获取的“第一条”数据是随机的，通常配合ORDER BY获取需要的第一条数据 | |

|  |  |  |  |
| --- | --- | --- | --- |
| （标题）**Filtering Data**  WHERE的使用 | | | |
| （概要）   1. WHERE与“条件”配合使用，“过滤”返回的数据 2. WHERE配合AND 3. WHERE配合OR 4. WHERE配合IN 5. WHERE配合LIKE 6. WHERE配合BETWEEN 7. WHERE配合<>使用 | （详情） | | |
| SELECT select\_list FROM table\_name WHERE condition; | | |
| SELECT first\_name FROM "customer" WHERE first\_name = 'Jamie' **AND** last\_name = 'Rice'; | | |
| SELECT first\_name FROM "customer" WHERE first\_name = 'Jamie' **OR** last\_name = 'Rice'; | | |
| SELECT first\_name, last\_name FROM customer WHERE first\_name **IN** ('Ann','Anne','Annie'); | | |
| SELECT first\_name, last\_name FROM customer WHERE first\_name **LIKE** 'An%'; | | |
| SELECT first\_name, "length"(first\_name) name\_length FROM customer WHERE "length"(first\_name) **BETWEEN** 3 AND 5; | | |
| SELECT first\_name, last\_name FROM customer WHERE first\_name LIKE'Bra%' AND last\_name **<>** 'Motley' ; | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
|  | | |
| （总结）  WHERE就是过滤条件，用来对指定的数据做处理操作。 | 常用“操作符” | = | Equal |
| > | Greater than |
| < | Less than |
| >= | Greater than or equal |
| <= | Less than or equal |
| <> or != | Not equal |
| AND | Logical operator AND |
| OR | Logical operator OR |

|  |  |
| --- | --- |
| （标题）**Filtering Data**  LIMIT的使用  可与OFFSET配合，与ORDER BY配合 | |
| （概要）   1. 与ORDER BY配合 2. 与OFFSET配合，跳过前几条数据，如果OFFSET的值比较大，那些数据同样会被计算（即使他们没有被返回） | （详情） |
| SELECT \* FROM film ORDER BY film\_id LIMIT 5; |
| SELECT \* FROM film ORDER BY film\_id LIMIT 5 OFFSET 3; |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结）  LIMIT常用于数据内容测试，检查数据表格中是否有数据。  与ORDER BY配合使用，帮助用户获取指定顺序的数据  与OFFSET配合，跳过指定数据的数据。 | |

|  |  |
| --- | --- |
| （标题）**Filtering Data**  FETCH的使用 | |
| （概要）   1. 获取第一行数据 2. 获取第一行数据 3. 获取前五行数据 4. 跳过5行获取前五行 | （详情） |
| SELECT \* FROM film ORDER BY title FETCH FIRST ROW ONLY; |
| SELECT \* FROM film ORDER BY title FETCH FIRST 1 ROW ONLY; |
| SELECT \* FROM film ORDER BY title FETCH FIRST 5 ROW ONLY; |
| SELECT \* FROM film ORDER BY title OFFSET 5 ROWS FETCH FIRST 5 ROW ONLY; |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结）   1. FETCH中 OFFSET使用与LIMIT类似 2. 参数FIRST与NEXT可相互替换 3. ROW与ROWS可相互替换 | |

|  |  |
| --- | --- |
| （标题）**Filtering Data**  IN 的使用 | |
| （概要）   1. IN 配合列表 2. IN配合“subquery” 3. NOT IN | （详情） |
| value IN (value1,value2,...)  SELECT \* FROM "rental" WHERE customer\_id IN (1,2); |
| value IN (SELECT value FROM tbl\_name);  SELECT \* FROM "rental" WHERE CAST(return\_date AS DATE) = '2005-06-03'; |
| SELECT \* FROM "rental" WHERE customer\_id NOT IN (1,2); |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结）   1. IN配合列表，等价于“equal (=) and OR operators”，但“IN”更快 2. NOT IN配合列表，等价于“not equal (<>) and the AND operators” 3. NOT IN配合“subquery”时候，如果“subquery”中包含“NULL”，则将出错或者速度很慢。 4. “CAST”是将一种datatype转化成另外一种。“CAST ( expression AS target\_type );” | |

|  |  |
| --- | --- |
| （标题）**Filtering Data**  WHERE的使用 | |
| （概要）  1、 | （详情） |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结） | |

|  |  |
| --- | --- |
| （标题）**Filtering Data**  WHERE的使用 | |
| （概要）  1、 | （详情） |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结） | |

|  |  |
| --- | --- |
| （标题）**Filtering Data**  WHERE的使用 | |
| （概要）  1、 | （详情） |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结） | |

|  |  |
| --- | --- |
| （标题）**Filtering Data**  WHERE的使用 | |
| （概要）  1、 | （详情） |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
| （总结） | |