数据库基础知识

介绍数据库基本知识以及SQL基本编写，主要以POSTGRESQL为例

数据库

- **数据库**指的是以**一定方式**储存在一起、能为多个用户共享、具有尽可能小的冗余度、与应用程序彼此独立的数据集合。

数据库管理系统

- 数据库管理系统（英语：Database Management System，简称DBMS）是为管理数据库而设计的电脑软件系统，一般具有存储、截取、安全保障、备份等基础功能。

关系型数据库

- 现代DBMS使用不同的数据库模型追踪实体、属性和关系。在个人电脑、大型计算机和主机上应用最广泛的数据库管理系统是关系型DBMS（relational DBMS）。
- 在关系型数据模型中，用二维表格表示数据库中的数据。这些表格称为关系。
- 关系实际上是*表*的数学称呼。
- 现在使用度较高的关系型数据库：Mysql , Oracle , SQL Server , DB2 , PostgreSQL

表

- 在关系数据库中，**数据库表**是一系列**二维数组**的集合，用来代表和储存数据对象之间的关系。它由纵向的**列**和横向的**行**组成。

- 每个表都是一个命名的行的集合。每一行由一组相同的命名 字段组成。而且每个字段都有一个特定的类型。

例子：表(table)

订单编号	订单类别	产品编号	数量	价格
20005	1	BR01	100	5.49
20005	2	BR03	100	10.99
20006	1	BR01	20	5.99
20006	2	BR02	10	8.99
20006	3	BR03	10	11.99
20007	1	BR03	50	11.49
20007	2	BNBG01	100	2.99
20007	3	BNBG02	100	2.99
20007	4	BNBG03	100	2.99
20007	5	RGAN01	50	4.49
20008	1	RGAN01	5	4.99
20008	2	BR03	5	11.99
20008	3	BNBG01	10	3.49
20008	4	BNBG02	10	3.49

什么是SQL？

- **结构化查询语言**（英语：Structural Query Language，缩写：**SQL**），是一种特殊目的之编程语言，**用于数据库中的标准数据查询语言**。
  - 不过各种通行的数据库系统在其实践过程中都对SQL规范作了某些编改和扩充。所以，**实际上不同数据库系统之间的SQL不能完全相互通用。**

SQL特点

- SQL是高级的非过程化编程语言，它允许用户在高层数据结构上工作。
- **以记录项目〔records〕的合集（set）〔项集，record set〕作为操纵对象，所有SQL语句接受项集作为输入，返回提交的项集作为输出。**
- 在多数情况下，在其他编程语言中需要用一大段程序才可实践的一个单独事件，而其在SQL上只需要一个语句就可以被表达出来。这也意味着用SQL可以写出非常复杂的语句。

SQL 特点，个人概括

- SQL操作的，永远都是数据集合。

SQL语句共分为3类

- 数据操纵语言（DML） Data Manipulation Language
  - INSERT、UPDATE、**SELECT**、DELETE
- 数据定义语言（DDL） Data Definition Language
  - CREATE、DROP 、ALTER
- 数据控制语言（DCL） Data Control Language
  - COMMIT 、ROLLBACK、GRANT、REVOKE

Q & A

- 视频主要内容：
- 只涉及到DML语句中SELECT 语句 操作，若对于除 SELECT 之外的其他内容有兴趣可以简单带过，除 SELECT 之外的太深入问题本视频不涉及。

视频使用数据介绍

表中文名	表英文名	字段英文名	字段中文名	数据类型
供应商表	Vendors	vend_id	供应商编号	char(10)
供应商表	Vendors	vend_name	供应商名称	char(50)
供应商表	Vendors	vend_address	供应商地址	char(50)
供应商表	Vendors	vend_city	供应商城市	char(50)
供应商表	Vendors	vend_state	供应商市级单位	char(5)
供应商表	Vendors	vend_zip	供应商邮编	char(10)
供应商表	Vendors	vend_country	供应商国家	char(50)

表中文名	表英文名	字段英文名	字段中文名	数据类型
产品信息表	Product	prod_id	产品编号	character(10)
产品信息表	Product	vend_id	供应商编号	character(10)
产品信息表	Product	prod_name	产品名称	character(255)
产品信息表	Product	prod_price	产品价格	numeric(8,2)
产品信息表	Product	prod_desc	产品描述	character varying(1000)

视频使用数据介绍（客户信息表）

表中文名	表英文名	字段英文名	字段中文名	数据类型
客户信息表	Customers	cust_id	客户编号	char(10)
客户信息表	Customers	cust_name	客户名称	char(50)
客户信息表	Customers	cust_address	客户地址	char(50)
客户信息表	Customers	cust_city	客户城市	char(50)
客户信息表	Customers	cust_state	客户市级单位	char(5)
客户信息表	Customers	cust_zip	客户邮编	char(10)
客户信息表	Customers	cust_country	客户国家	char(50)
客户信息表	Customers	cust_contact	客户联系名	char(50)
客户信息表	Customers	cust_email	客户邮件	char(255)

视频使用数据介绍

表中文名	表英文名	字段英文名	字段中文名	数据类型
订单表	Orders	order_num	订单号	int
订单表	Orders	order_date	订单日期	date
订单表	Orders	cust_id	客户编号	char(10)

表中文名	表英文名	字段英文名	字段中文名	数据类型
订单内容表	OrderItems	order_num	订单号	int
订单内容表	OrderItems	order_item	订单内物品编号	int
订单内容表	OrderItems	prod_id	产品编号	char(10)
订单内容表	OrderItems	quantity	物品数量	int
订单内容表	OrderItems	item_price	物品价格	decimal(8,2)

查询概述

- 从数据库中检索数据的过程或命令叫做*查询*。
- 在 SQL 里SELECT命令用于声明查询。

基本语法：

SELECT 列名,.... FROM 表名；

返回部分数据（部分列）

- 查看订单表
  - SELECT prod_id, vend_id, prod_price FROM products;
  - 语句含义：查看订单表中所有字段

  - SQL中使用空白字符分割语句
  - Select 表示查询
  - prod_id, vend_id, prod_price 列名
  - From 数据来源表
  - 分号是SQL语句的语句结尾符号

补充知识:

- 多字段用逗号分割
- * 号是通配符，表示所有字段
- 使用 AS 设置别名
- SQL 语法中大小写不敏感。*大小写敏感需要转义。*
- DISTINCT 去重

返回部分数据（部分行）

- select * from products limit 5 ;
- 含义：返回 product 在从硬盘上找到的前五条数据

- PG(PostgreSQL 简称PG) 中限制返回条目使用limit 关键字
- SQL server 中使用 TOP
- Oracle 使用 where rownum < 5

- select * from product limit 2 offset 3;
- 含义：返回从第 3 行开始的 2 行数据

返回部分数据

- 样例需求：
  - 查看商品价格最小的说那个产品
  - select * from products order by prod_price limit 3 ;

- 怎么理解这个SQL在数据库中怎么抽取并处理数据的呢？

SELECT 语句 执行顺序

- 第一步：执行 FROM 子句， 从 products 表组装数据源的数据
- 第二步：执行 WHERE 子句，本例不含有WHERE字句
- 第三步：执行 GROUP BY 子句，本例不含有 GROUP BY 字句
- 第四步：执行 HAVING 子句，本例不含有 HAVING 字句
- 第五步：执行 SELECT,这个阶段是投影的过程，处理 SELECT 子句提到的元素
- 第六步：执行 ORDER BY 子句，以订单编号从小到大排列
- 第七步：执行LIMIT [OFFSET]子句，返回最前面三条语句

FROM

- FROM 才是 SQL 语句执行的第一步，并非 SELECT 。数据库在执行 SQL 语句的第一步是将数据从硬盘加载到数据缓冲区中，以便对这些数据进行操作。

SELECT

- SELECT 是在大部分语句执行了之后才执行的。
- 严格的说是在 FROM 和 GROUP BY 之后执行的。理解这一点是非常重要的，这就是你不能在 WHERE 中使用在 SELECT 中设定别名的字段作为判断条件的原因。

ORDER BY

- ORDER BY 默认升序，升序关键字（ASC），降序排列关键字（DESC）
- ORDER BY 多个字段，执行顺序从左至右
- ORDER BY多个字段可以使用不同的顺序

- **ORDER BY 需要把所有符合条件的数据全部加载至内存然后进行排序，再显示输出，在数据量过大的情况下，性能下降的非常快。**

因为是在内存内排序，所以占用大量内存后会导致性能下降。没必要的要不要用order by

LIMIT

- 限制输出
- 各具灵活性的限制输出

数据过滤，WHERE 子句

- < 小于
- > 大于
- = 等于
- <= 小于等于
- >= 大于等于
- <> , != 不等于

- a BETWEEN x AND y
  - 等效于 a >= x AND a <= y
- a NOT BETWEEN x AND y
  - 等效于 a < x AND a > y
- NULL
  - expression IS NULL
  - expression IS NOT NULL
  - 错误：expression = NULL

检查单个值

- SELECT prod_name, prod_price FROM Products
- WHERE prod_price < 10;

- 含义：查询所有价格小于10 的商品

- 把上个例子中的order by limit 3 复制过来

- SELECT prod_name, prod_price FROM Products
- WHERE prod_price < 10; order by prod_price limit 3 ;
- 执行顺序解释

WHERE 子句的具体执行逻辑

- 物品价格小于 10 在数据库里面是怎么执行的？
- 在人处理数据的过程中，是逐条记录比较，程序实现是怎样的？
- 本质上，数据库进行了逐条扫描数据，并与50做比对，符合条件返回True，不符合返回False

多个 WHERE 子句

- 价格大于 10 并且是来自 供应商 ‘BRS01’ 的商品
- 逻辑运算：与 、或、非
  - AND , OR , NOT

  AND OR 的优先级

  AND > OR
- IN 操作符
  - IN 可以与子查询一起工作

子查询

- 需求：列出订购物品 RGAN01 的所有顾客
- 不使用子查询
  - 首先：找出所有包含 物品RGA01 的订单
  - SELECT order_num FROM OrderItems WHERE prod_id = 'RGAN01';
  - 其实：依据找到的订单编号找到所有的顾客
  - SELECT cust_id FROM Orders WHERE order_num IN (20007,20008);
- 使用子查询

LIKE关键字

- LIKE是一种模糊匹配
- 最常见的通配符是%，表示任意数量的任意字符
- select * from products where vend_id like 'BR%' ;

%是like的专用语句，通配符，代表任意数量的任意字符

分组聚合

- 简单的说，聚合函数是按照一定的规则将多行数据**汇总成一行**的函数。

- 对数据进行聚合前，还可以按照特定的列将数据进行分组(Group by)再聚合,然后按照再次给定的条件进行筛选(Having).

  having 仅用于 group by。而where是在group by之前执行的

聚合

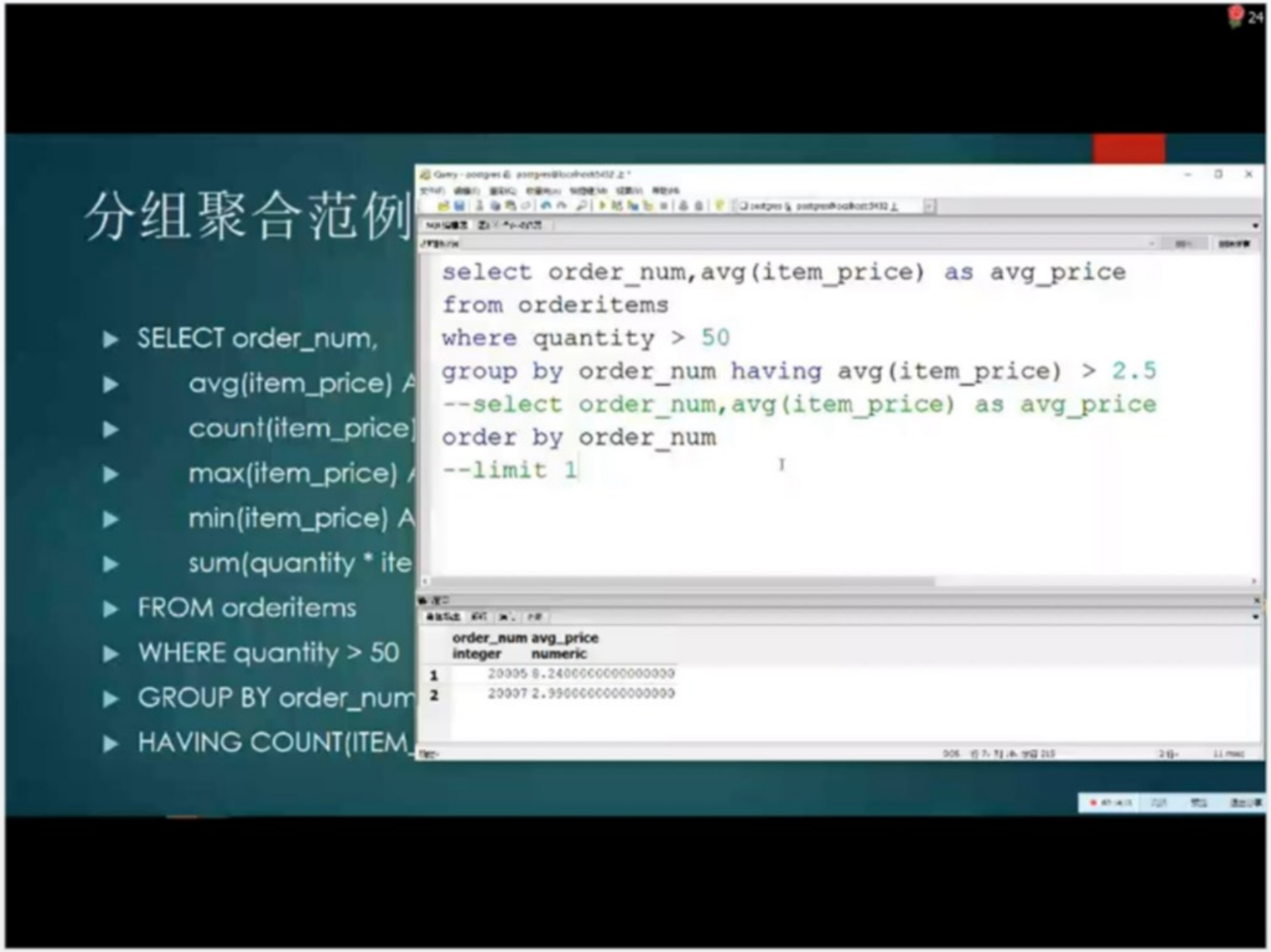
- 函数
  - AVG 均值
  - COUNT 非空行数
  - MAX 最大值
  - MIN 最小值
  - SUM 求和

分组

- 如果聚合函数所得到的结果无法按照特定的值进行分组，那聚合函数的作用就没那么强了。
- 在SQL中，使用**Group by**对聚合函数汇总的值进行分组。

当前状态：仅管理员可以上台

管理



值，他会怎么选择出现的order number的值啊？

古董(923603507) 21:02:33
having是备胎

沉淀(821915982) 21:02:37

普通朋友a<nickzhu85@gmail.com> 21:02:38
应该事后再练练 呵呵

Jason517(415229574) 21:02:39
结合看书

大狼(812913245) 21:02:42
就是因为groupby后面不能接where 所以才有了having这个语句

谷子雨(2538740718) 21:02:46
having 就是为group by 而生的

Dennis王锐程(171628155) 21:02:46
Having相当于只能针对group里面用的where

露露酱(760138051) 21:02:49
讲的很好

乔(79711070) 21:02:51
分组 + 聚合 是一起的，

库存： x0 购买

送给: 沈(454900743) 数量: 9 赠送

按住F2说话 播放伴奏 上台

[广告]一分钟决定租的房 发送

分组聚合范例：

分组聚合函数值得就是group by和having，但是如果想在where里放一些分组聚合的条件，是会出错的。而having就是专门干这个的，就只干这么一个而已

```
SELECT order_num,
       avg(item_price) AS avg_price,
       count(item_price) AS cnt_price,
       max(item_price) AS max_price,
       min(item_price) AS min_price,
       sum(quantity * item_price) AS order_price
FROM orderitems
WHERE quantity > 50
GROUP BY order_num
HAVING COUNT(ITEM_PRICE) > 2
```

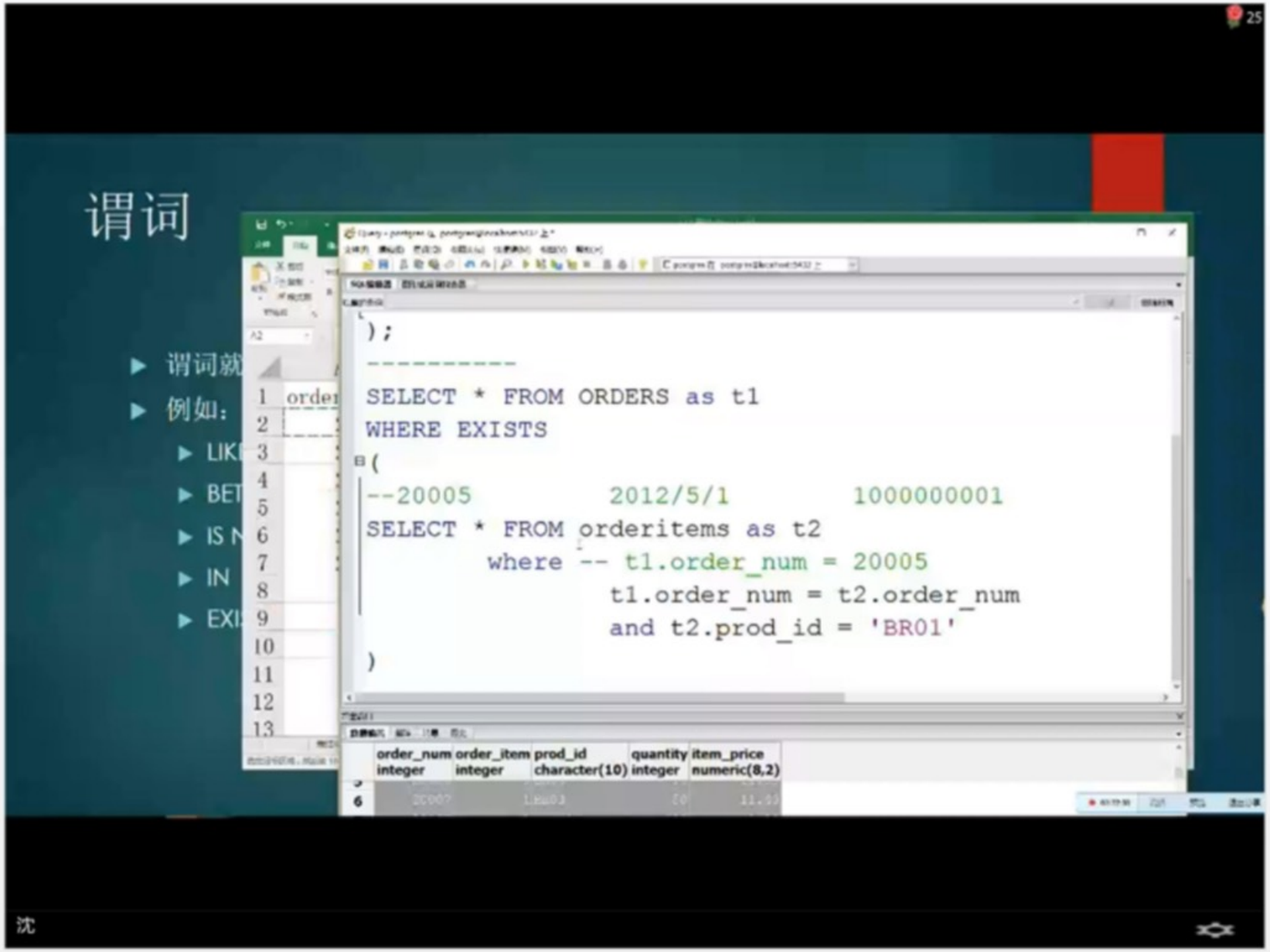
其它函数

- 可以应用在 SQL 语句中的函数
- 可分为四类
  - 数值处理（也称为数学函数）
  - 字符处理
  - 日期处理
  - 类型转换

谓词

- 谓词就是返回值为布尔值得函数
- 例如：
  - LIKE
  - BETWEEN
  - IS NULL，IS NOT NULI
  - IN
  - EXISTS

第一个where会把得到的数据逐个传入到exists里，让第二个where进行逐个判断



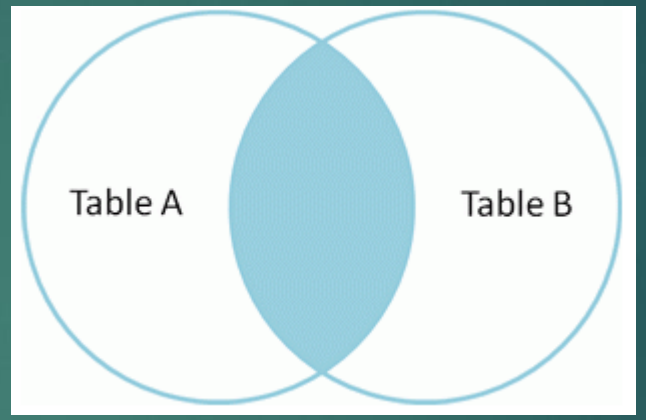
杭州—MAOn(958093654) 21:11:52
已经看晕了。。
小祥(920082975) 21:12:31
exists是比较难理解的。。。
Dennis王锐程(171628155) 21:12:38
.是不是“的”的意思，t1的orderitem那一列？
喜宝(759713802) 21:12:39
不能执行实验正确性吗
单人旅途(345686985) 21:13:24
excel放大一点吧
Dennis王锐程(171628155) 21:15:59
大概懂了= =
单人旅途(345686985) 21:16:11
第一遍的循环吗是20005的是么？
cairrot(1121053890) 21:16:14
似乎懂了
王炳明(942663728) 21:16:21
明白
Dennis王锐程(171628155) 21:16:34
en

表的几种JOIN

- 为什么使用JOIN?
  - 希望将数据分开存放，避免过多的重复数据
  - 避免数据变更的时候，需要更新太多数据

表的几种JOIN

Inner join
产生的结果集中，是A和B的交集。



SELECT * FROM TableA

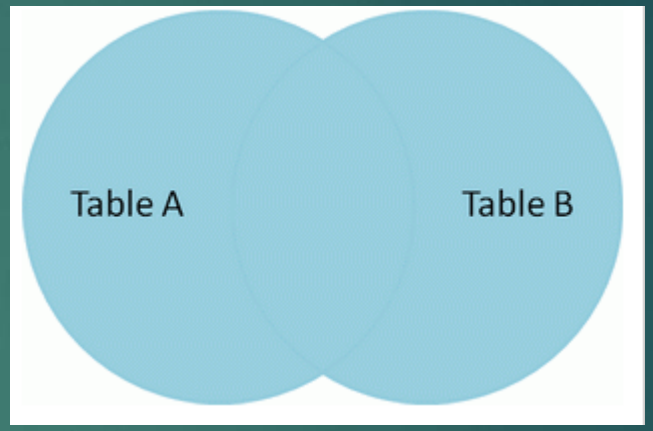
INNER JOIN TableB

ON TableA.name = TableB.name

例子：

- SELECT vend_name, prod_name, prod_price FROM Vendors **inner join** Products ON Vendors.vend_id = Products.vend_id;

表的几种JOIN



Full outer join

产生A和B的并集。但是需要注意的是，对于没有匹配的记录，则会以null做为值。

SELECT * FROM TableA

FULL OUTER JOIN TableB

ON TableA.name = TableB.name

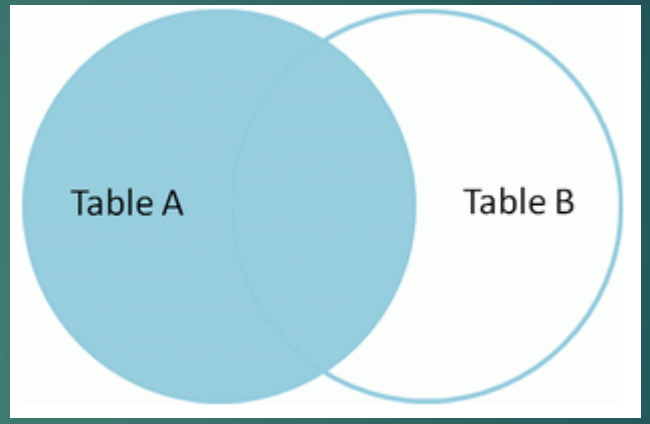
例：

- SELECT Customers.cust_id, Orders.order_num
  FROM Customers FULL OUTER JOIN Orders
  ON Customers.cust_id = Orders.cust_id;

表的几种JOIN

Left outer join

产生表A的完全集，而B表中匹配的则有值，没有匹配的则以null值取代。




```
SELECT * FROM TableA
LEFT OUTER JOIN TableB
ON TableA.name = TableB.name
```

例：

- SELECT Customers.cust_id, Orders.order_num
  FROM Customers LEFT OUTER JOIN Orders
  ON Customers.cust_id = Orders.cust_id;

表的几种JOIN

cross join

这种Join没有办法用文式图表示，因为其就是把表A和表B的数据进行一个N*M的组合，即笛卡尔积。

SELECT * FROM TableA

CROSS JOIN TableB ;

例：

- SELECT vend_name, prod_name, prod_price
  FROM Vendors cross join Products;