

数据库基础

(PDF测试岗位课程)

SUNING 苏宁



数据库基础知识

数据库常用操作

常见报错及解决方法

数据恢复及备份

课程回顾, Q&A

数据：指描述事物的符号记录。数据与语义(含义)密不可分。

举例：数字、文字、图形、图像、声音、语言等。

例如：(201505101, 张三, 男, 21, 上海, 计算机专业)

信息：经加工处理后获取的有价值的数据

数据与信息的关系？
 $\text{信息} = \text{数据} + \text{数据处理}$

数据处理：指对各种数据进行收集、存储、加工和传播的一系列活动总和。

数据管理：指对数据进行分类、组织、编码、存储、检索和维护，它是数据处理的中心问题。

常用术语:

数据项 (Field) : 标记实体属性的符号集, 数据库中可命名及存取的最小逻辑数据单位。例如, 学生有学号、姓名、年龄、性别等字段。

记录 (Record) : 字段的有序集合称为记录。一般用一个记录描述一个实体, 例如, 一个学生记录由有序的字集组成: (学号, 姓名, 年龄, 性别)。

文件 (File) : 同一类记录的汇集称为文件。文件是描述实体集的, 例如, 所有学生记录组成了一个学生文件。

码 (Key) : 能惟一标识文件中每个记录的字段或字段集, 称为记录的码。这个概念与实体的码相对应。例如, 学生的学号可以作为学生记录的码(注: 关系数据库系统中称为元组)

员工基本信息表

字段、属性
----列

| 学 号 | 姓 名 | 年 龄 | 性 别 | 体 系 | 职级 |
|----------|-----|-----|-----|-----|-----|
| 13061234 | 王小明 | 25 | 女 | 信息 | B6 |
| 13061235 | 黄大鹏 | 27 | 男 | 金融 | B7 |
| 13070625 | 张文斌 | 28 | 男 | 置业 | L6 |
| ... | ... | ... | ... | ... | ... |

记录、元组
----行

数据管理技术的三个阶段

SUNING 苏宁



| | | 人工管理阶段 | 文件系统阶段 | 数据库系统阶段 |
|----|---------|-------------|--------------|---------------------|
| 背景 | 应用背景 | 科学计算 | 科学计算、管理 | 大规模管理 |
| | 硬件背景 | 无直接存取存储设备 | 磁盘 | 大容量磁盘 |
| | 软件背景 | 没有操作系统 | 有文件系统 | 有DBMS |
| | 处理方式 | 批处理 | 实时处理、批处理 | 实时处理、分布处理、批处理 |
| 特点 | 数据的管理者 | 用户（程序员） | 文件系统 | 数据库管理系统 |
| | 数据面向的对象 | 某一应用程序 | 某一应用 | 现实世界 |
| | 数据的共享程度 | 无共享，冗余度很大 | 共享性差，冗余度大 | 共享性高，冗余度小 |
| | 数据的独立性 | 不独立，完全依赖于程序 | 独立性差 | 具有高度的物理独立性和一定的逻辑独立性 |
| | 数据的结构化 | 无结构 | 记录内有结构、整体无结构 | 整体结构化，用数据模型描述 |
| | 数据控制能力 | 应用程序自己控制 | 文件系统 | 数据库管理系统 |

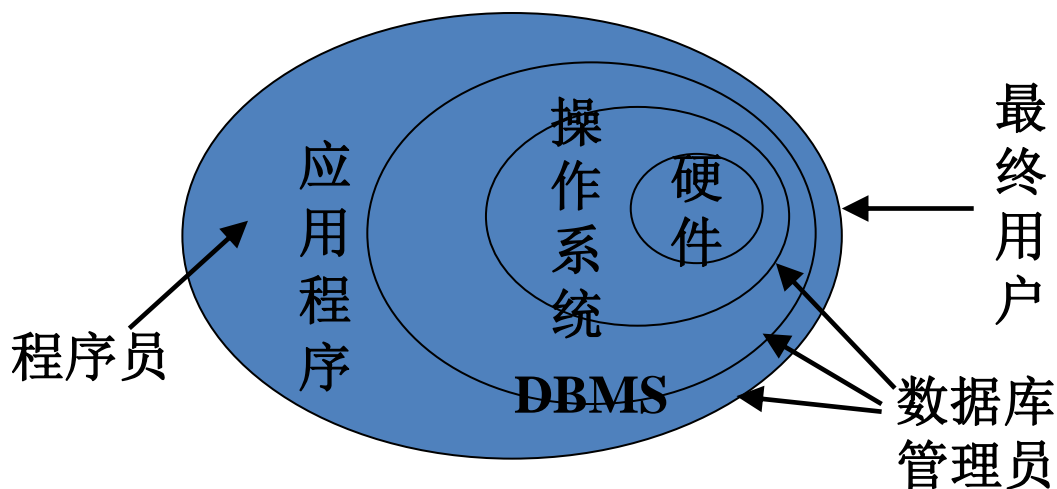
(1) 数据库 (DataBase, 简称 DB)

定义：长期存放在计算机存储设备上有组织的、可共享的、相关的数据集合。

特征：数据按一定的数据模型组织、描述和储存；可为各种用户共享；冗余度较小；数据独立性较高；易扩展。

(2) 数据库管理系统 (DataBase Management System, 简称DBMS)

定义：负责对DB进行管理与维护。



数据库系统层次示意图

数据库系统（DataBase System，简称DBS）

即具有管理和控制数据库功能的计算机系统。

数据库系统的5个组成部分：

- ◆ 硬件系统
- ◆ 数据库DB
- ◆ 数据库管理系统DBMS
- ◆ 相关软件（支持软件，如操作系统等；应用开发工具软件等）
- ◆ 数据库系统用户：
 - ▼ 数据库管理员（DBA）：负责数据库管理的和维护操作。

数据定义功能：数据定义语言（Data Definition Language, 简称DDL）

如：新建表，修改表，删除表等操作

测试人员需要
重点掌握DDL
和DML

数据操纵功能：数据操纵语言（Data Manipulation Language, 简称DML）

如：插入、查找、修改和删除数据等操作

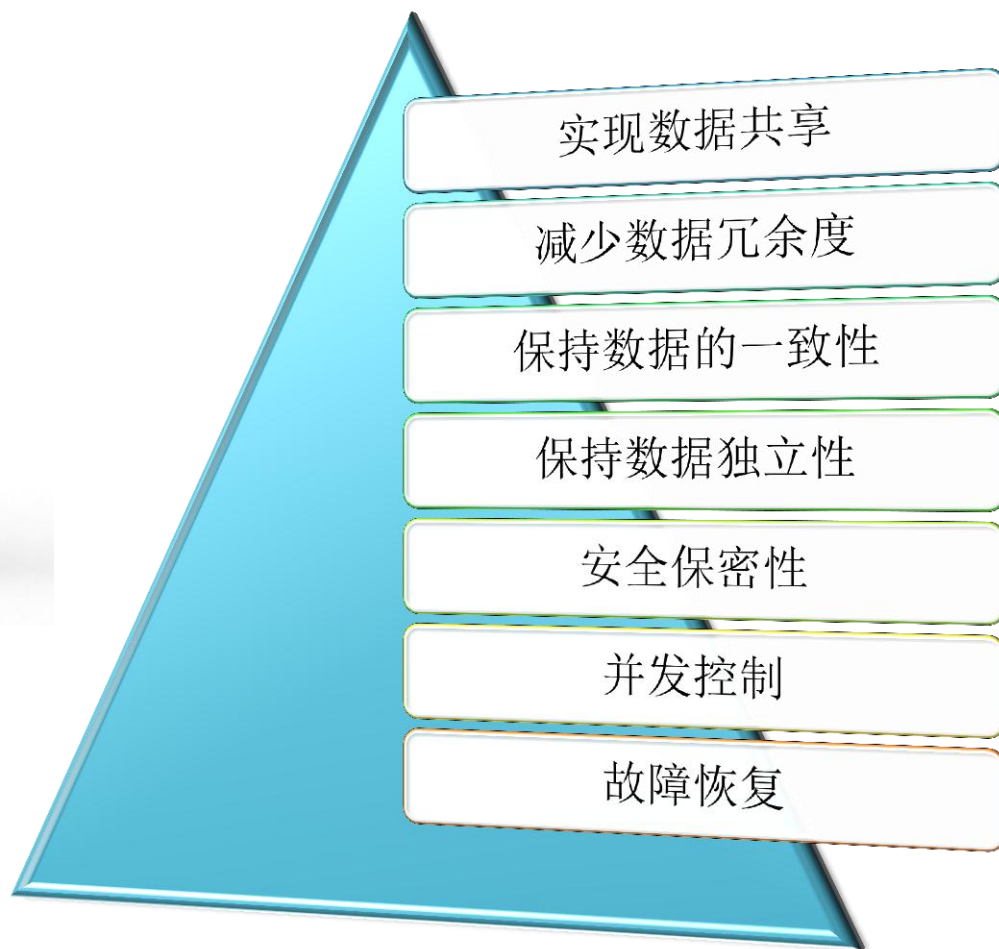
数据库的运行管理：数据控制语言（Data Control Language, 简称 DCL）, 如：commit确认，rollback取消。统一管理及控制，以保证数据的安全性，完整性，共享性及可恢复性

数据库的建立和维护功能：用户赋权，收回权限，数据的输入，转换，转储，恢复，监控，分析等

数据安全性保护：数据库登录的身份验证管理，数据库的使用权限管理和数据库中对象的使用权限管理等

数据完整性检查：一般有实体完整性约束、参照完整性约束、函数依赖约束、统计约束四类，用来保证数据的正确性、有效性、一致性

并发控制：最常用的并发控制技术是加锁技术，实现多个用户同时对数据库中数据或同一数据进行合理存取



模式 (Schema)：是数据库中全体数据的逻辑结构和特征的描述。（不涉及到具体的值）

例：员工基本信息数据库模式：

员工（工号，姓名，性别，体系，职级）

体系（信息，置业，酒店等）

职级（B6，B7，L1，L2…）

注意：模式中不包含具体的记录数据

模式的实例：模式的具体值称为模式的实例。



模式与实例的区别：

- 1) 模式相对稳定；
- 2) 同一个模式，可以有很多实例，实例是相对变动的；
- 3) 模式反映的是数据结构及其联系，而实例反映的是数据库某一时刻的状态值；

- 概念模型
 - 也称信息模型，按用户的观点来对数据和信息建模
 - 用于数据库设计
- 数据模型
 - 按计算机的观点对数据建模，主要用于DBMS的实现
 - 一般有层次，网状，关系三种
- 客观对象的抽象过程---两步抽象
 - 现实世界中的客观对象抽象为概念模型；
 - 把概念模型转换为某一DBMS支持的数据模型
- 概念模型是现实世界到机器世界的一个中间层次



- **数据结构**
 - 指所研究的对象类型的集合
 - 与数据类型、内容、性质有关的对象
 - 与数据之间联系有关的对象
 - 是对系统静态特性的描述
- **数据操作**
 - 对数据库中各种对象（型）的实例（值）允许执行的操作及有关的操作规则
 - 检索、更新（包括插入、删除、修改）
 - 是对系统动态特性的描述
- **数据的约束条件**
 - 一组完整性规则的集合
 - 是数据及其联系所具有的制约和储存规则，用以限定符合数据模型的数据库状态以及状态的变化，以保证数据的正确、有效、相容

送达方信息（会员编号，联系电话，地址信息，商品名称）
物流明细信息（姓名，身份证号，联系电话，预计出库时间，期望送达时间）

用户基本信息（会员编号，姓名，身份证号，联系电话，送货地址）
送达方信息（会员编号，联系电话，地址信息，商品名称）

订单交易系统

订单执行系统

订单查询系统

...

例：

用户基本信息（会员编号，姓名，身份证号，联系电话，送货地址）
送达方信息（会员编号，联系电话，地址信息，商品名称）

后端某系统（如LES）

用户基本信息（会员编号，姓名，身份证号，联系电话，送货地址）
送达方信息（会员编号，联系电话，地址信息，商品名称）
物流明细信息（姓名，身份证号，联系电话，预计出库时间，期望送达时间）

思考：1、交易系统用户基本信息表中增加一个字段，对其他3个系统有影响吗？
2、交易系统用户基本信息表将字段名“联系电话”改名为“移动电话”，对3个系统有影响吗？

数据库文件的存储形式，如文件名，是否索引等。

数据库的物理存储模式

数据库基础知识

数据库常用操作

常见报错及解决方法

数据恢复及备份

课程回顾, Q&A

CREATE语法

- CREATE TABLE语法
- CREATE INDEX语法

DROP语法

- DROP DATABASE语法
- DROP INDEX语法
- DROP TABLE语法

ALTER TABLE语法

- ALTER TABLE语法

- 常见表级完整性约束：
 - PRIMARY KEY 主键约束
 - UNIQUE 唯一性约束
 - NOT NULL 非空值约束
 - AUTO_INCREMENT 用于整数列默认自增1
 - UNSIGNED 无符号整数
 - DEFAULT default_value 默认值约束
 - DEFAULT cur_timestamp 创建新记录时默认保存当前时间
(仅适用timestamp数据列)
 - ON UPDATE cur_timestamp 修改记录时默认保存当前时间 (仅适用timestamp数据列)

- 数据表之间的关联/引用关系是依靠具体的主键（primary key）和外键（foreign key）建立起来的。

主键：帮助数据库以最快速度把一条特点数据记录的位置确定下来。

主键必须是唯一的

主键应该是紧凑的，因此整数类型比较适合

外键：引用另外一个数据表的某条记录。

外键列类型尽可能与主键列类型保持一致

外键列应该加上NOT NULL

举例：

主键

```
create table student(  
    sid int not null auto_increment,  
    name varchar(20) not null,  
    primary key(sid)  
);
```

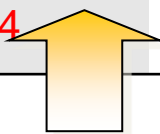
外键（自动检查外键是否匹配，仅适用InnoDB）

```
create table score(  
    cid int not null auto_increment  
primary key,  
    score int,  
    sid int,  
    foreign key(sid) references  
student(sid)  
);
```

| 学号 | 姓名 | 地址 | |
|---------|-----|------|------|
| 0010012 | 李山 | 山东定陶 | |
| 0010013 | 吴兰 | 湖南新田 | |
| 0010014 | 雷铜 | 江西南昌 | |
| 0010015 | 张丽鹃 | 河南新乡 | |
| 0010016 | 赵可以 | 河南新乡 | |

✕

0010014



雷铜

江西南昌

- 举例：Create table products(
id char(20) not null,
price float default 0.01,
constraint primary key (id)
);

约束方法：唯一约束、主键约束、标识列

约束方法：限制数据类型、检查约束、外键约束、默认值、非空约束

AUTO_INCREMENT

- 自动标识列，在需要产生唯一标志符号或者顺序值时候，可用此属性。值一般从1开始，每行增加1，在插入NULL到一个AUTO_INCREMENT列时，数据库会插入一个比该列中当前最大值大1 的值，一个表中最多能有一个有此属性的列。对于想使用此属性的列应该定义为NOT NULL,并定义为PRIMARY KEY 或者定义为UNIQUE键。

- `create table t(id int auto_increment not null primary key);`

UNSIGNED

- 无符号，此属性禁用负值，将数值的取值范围从零开始。

- `create table t(num tinyint , num2 tinyint unsigned);`

NULL和NOT NULL

- 默认为NULL，即插入值时没有在此字段插入值时自动填NULL，如果指定了NOT NULL，则必须在插入值时在此字段添入值，不允许插入NULL值

DEFAULT

- DEFAULT 约束用于向列中插入默认值，如果没有规定其他的值，那么会将默认值添加到所有的新记录。
- Create table Persons
(
P_Id int NOT NULL,
Name varchar(255) NOT NULL,
City varchar(255) DEFAULT 'Sandnes'
)

- **CREATE DATABASE**语法

说明：**CREATE DATABASE**用于创建数据库，进行命名。如果要使用**CREATE DATABASE**，您需要获得数据库**CREATE**权限；

语法：**CREATE DATABASE** *db_name* ;

举例：**Create Database test1;**

- **CREATE TABLE**语法

说明：在当前数据库下新创建一个数据表

语法：**create table** 表名 (

列名1 列类型 [**<列的完整性约束>**],

列名2 列类型 [**<列的完整性约束>**],

... ...);

举例：建立一个表**school**,其由两列组成，第一列属性为非空，并做为主键,并自增

create table school(school_id int(10) not null auto_increment primary key,
school_name varchar(20));（注：标红的三个关键字为约束条件，后续页会重点介绍）

- 删除表操作

语法: `drop table [if exists] tab_name [,tab_name]...`

叙述: 从数据库中删除给定的表。如果给出**if exists** 子句, 则删除不存在的表不会出错。

- 删除数据库操作

语法: `drop database [if exists] db_name`

叙述: 删除给定的数据库。在删除一个数据库后, 它就永远没有了, 因此要特别小心。如果给出**if exists** 子句, 则删除不存在的数据库不会出错。



从删除内容上区分

- truncate和 delete只删除数据不删除表的结构(定义);
drop语句将删除表的结构、被依赖的约束(constrain), 触发器(trigger), 索引(index); 依赖于该表的存储过程/函数将保留, 但是变为invalid状态;

从语句类型上区分

- delete语句是dml, 这个操作会放到rollback segment中, 事务提交之后才生效; 如果有相应的trigger, 执行的时候将被触发;
truncate, drop是ddl, 操作立即生效, 原数据不放到rollback segment中, 不能回滚, 操作不触发trigger;

从对表空间的影响来区分

- delete语句不影响表所占用的区间, 高水线(high watermark)保持原位置不动;
drop语句将表所占用的空间全部释放;
truncate语句缺省情况下将空间释放到minextents个extent, 除非使用reuse storage; 而且truncate会将高水线复位(回到最开始);

从删除速度来看

- drop > truncate > delete

- 语法: **alter table** 表名 **action**;
- 说明: **action** 可以是如下语句:
 - **add** 列名 建表语句 [**first** | **after** 列名]
可以为表添加一列, 如果没指定**first**或者**after**, 则在列尾添加一列, 否则在指定列添加新列
 - **add primary key** (列名)
为表添加一个主键, 如果主键已经存在, 则出现错误
 - **add foreign key**(列名) **references** 表名(列名)
为表添加一个外键
 - **alter** 列名 **set default** 默认值
可以更改指定列默认值
- 语法: **change** 旧列名 新列名 <建表语句> [**first** | **after** 列名]
可以更改列类型和列名称, 如果原列的名字和新列的名字相同
modify 列名 <建表语句> [**first** | **after** 列名] //和**change**的作用相同
drop 列名 //可以删除一列
drop primary key //可以删除主键
engine 类型名 //可以改变表类型
rename as 新表名 //可以将表名更改

1、向people表中添加字段address2, 类型为varchar, 最大长度为100

- `alter table people add address2 varchar(100);`

2、将people表中的name列默认值改为100

- `alter table people alter name set default 100;`

3、向student表增加“入学时间”列，其数据类型为日期型

- `alter table student add scome date;`

注：无论基本表中原来是否已有数据，新增加的列一律为空值。

4、将Student表中年龄的数据类型改为半字长整数

- `alter table student modify sage smallint;`

- 注：修改原有的列定义有可能会破坏已有数据

- 创建视图语法：
Create view <视图名>(
 <列名 1 >,[<列名 2 >,....])
 As [select查询语句]
- 删除视图语法：
Drop view <视图名>
- 示例：
Create view stu_test(id,name)
As select stu_id,name from student;

Drop view stu_test;



数据操作语句重点介绍4个常用的：

- ✓ Select
- ✓ Update
- ✓ Insert
- ✓ Delete

Select语法：



SELECT [ALL|DISTINCT] <目标列表表达式>

[,<目标列表表达式>]...

FROM<表名或视图名>,[<表名或视图名>]...

[**WHERE**<条件表达式>]

[**GROUP BY** <列名1>[**HAVING**<条件表达式>]]

[**ORDER BY**<列名2>[ASC|DESC]];

说明：



SELECT：指明需要查询的项目，一般是列名，也可以是表达式，查询表中未存储但可导出的结果。

WHERE：用来说明查询的条件

DISTINCT：任选项用来消除查询结果的重复项

GROUP：用来将结果按列名1的值进行分组，该属性列值相等元组为一组

ORDER：将结果表按列名2的值升序或降序排序

where 子句用来查询满足条件的元组，如：

- ✓ 比较大小 (= , <=> , <= , >= , <>)
- ✓ 确定范围 (BETWEEN...AND/Not Between....and)
- ✓ 确定集合 (IN, NOT IN)
- ✓ 字符匹配 (LIKE , NOT LIKE)
- ✓ 逻辑运算符 (AND,OR,NOT)



• 举例：

Select * from student where scome_date > '20150901'; (* 单引号转义)

Select * from student where age not between 18 and 19;

Select * from student where name like 'a%'; (* 百分号转义)

Select * from student where stu_id in (select stu_id from score)

多表联接查询实际上是通过各个表之间共同列的关联性来查询数据，使用时应该注意以下几点：

- 1.尽可能以表的**主键和外键**来指定联接条件，如果以各表共同拥有的列来指定联接，则这些列的数据类型必须相同或相兼容。
- 2.如果所要联接的表具有相同的列名，则在引用这些列的时候，必须指定其表名，格式为：表名.列名
- 3.参与联接查询的表的个数越多，SQL SERVER处理这个查询所花费的时间就越长，所以应该尽量限制表的个数。
- 4.当表的名称太长时，为了书写上的方便简单，在SQL语言的查询语句中可以定义表的别名。



内连接

- 内连接 (Inner Join) 也叫自然连接, 使用JOIN或INNER JOIN关键字表示。在内连接中, 只有在两个表中匹配的行才能在结果集中出现。
- 通常有3种:
 1. 等值连接
 2. 自然连接
 3. 不等值连接
- 一. 等值连接

在连接条件中使用等号(=)运算符比较表中的列值. 在select子句中并没指定所需显示的列, 因此连接的列会显示两次, 产生冗余列.
- 【例】查询所有参加考试的学生基本信息和成绩信息

```
select * from student inner join score on student.sno=score.sno
```

二. 自然连接

在连接条件中使用等号(=)运算符比较被连接列的列值. 但他使用选择列表指定查询结果集合中所包括的列, 并删除连接表中的重复列.

```
select s.*, cno, grade from student s inner join score sc on s.sno=sc.sno
```

三. 不等值连接

在连接条件中使用除等号(=)运算符以外的其他比较运算符, 比较被连接的列的列值.

- A. 在where子句中定义连接

- 【例】查询每个学生的学号, 姓名, 邮编等基本信息及其所选课程的成绩情况:

```
select student.sno, sname, spostno, cno, grade
from student, score
where student.sno=score.sno
```

- B. 在from子句中定义连接

具体分为以下几种:

- **INNER JOIN** : 显示符合条件的记录, 此为默认值, 可省略inner; (内连接)
- **LEFT (OUTER) JOIN**: 显示符合条件的数据行以及左边表中不符合条件的数据行, 此时右边数据行会以NULL来显示, 此称为左连接;
- **RIGHT (OUTER) JOIN**: 显示符合条件的数据行以及右边表中不符合条件的数据行, 此时左边数据行会以NULL来显示, 此称为右连接;
- **FULL (OUTER) JOIN**: 显示符合条件的数据行以及左边表和右边表中不符合条件的数据行, 此时缺乏数据的数据行会以NULL来显示; (全连接)
- **CROSS JOIN**: 会将一个表的每一笔数据和另一表的每笔数据匹配成新的数据行。(交叉连接)
- 当将JOIN 关键词放于FROM子句中时, 应有关键词ON与之相对应, 以表明连接的条件。

1、左外连接：LEFT OUTER JOIN ON

- `SELECT e.last_name, e.department_id, d.department_name FROM employees e LEFT OUTER JOIN departments d ON (e.department_id = d.department_id);` 等价于
- `SELECT e.last_name, e.department_id, d.department_name FROM employees e, departments d WHERE e.department_id=d.department_id(+)`
- 结果为：所有员工及对应部门的记录，包括没有对应部门编号department_id的员工记录。

2、右连接：RIGHT OUTER JOIN ON

- `SELECT e.last_name, e.department_id, d.department_name FROM employees e RIGHT OUTER JOIN departments d ON (e.department_id = d.department_id);`
- 等价于 `SELECT e.last_name, e.department_id, d.department_name FROM employees e, departments d WHERE e.department_id(+)=d.department_id`
- 结果为：所有员工及对应部门的记录，包括没有任何员工的部门记录。

3、全连接：FULL OUTER JOIN

- `SELECT e.last_name, e.department_id, d.department_name FROM employees e FULL OUTER JOIN departments d ON (e.department_id = d.department_id);`
- 结果为：所有员工及对应部门的记录，包括没有对应部门编号department_id的员工记录和没有任何员工的部门记录。

- 语法: UPDATE <表名> SET <列名 = 更新值>

[WHERE <更新条件>]

说明: Where 子句是判断语句, 用来设定条件, 限制只更新匹配的行, 如果不带 where 子句, 则更新所有行数据。

举例:

```
Update student set sname='Tom' where sname='Alex';
```

```
Update Student set Sdept ='家政系' where Sdept = '刺绣系';
```

```
Update Score
```

```
Set Grade = Grade + 5
```

```
Where Grade <= 95 ;
```

- 语法: **INSERT [INTO] <表名> [列名] VALUES <值列表>**
- 说明: 如果表名后面没写字段名, 则默认是向所有的字段添加值, 另外字符串值应该用 **' '** 引号括起来
- 举例: 向people表中添加一条记录:

```
insert into people(name,age)
values('zhangsan', 20);
```

- 示例:

```
insert into Students (SName,SAddress,SGrade,SEmail,SSEX)
values ('小明');
```

这个语句错
在哪?

每次插入一行数据, 不可能只插入半行或者几列数据, 因此, 插入的数据是否有效将按照整行的完整性的要求来检验;

```
insert into Students (SName,SAddress,SGrade,SEmail,SSEX)
values ('张青山','上海松江',ZQS,'ZQS@Sohu.com', '男')
```

这个语句错
在哪个字段?

每个数据值的数据类型、精度和小数位数必须与相应的列匹配;

- 语法: **DELETE FROM <表名> [WHERE <删除条件>]**

说明: 此语句会删除表中的行, 如果不带where子句, 则删除整个表中的记录, 但是表不被删除。

举例:

- ✓ 删除student表中的所有年龄小于18岁的记录:

```
Delete from student where sage <18;
```

- ✓ 删除学生编号为'95005'的学生信息包括成绩信息

```
Delete from student,sc
```

```
using student,sc
```

```
where student.sno = sc.studentid
```

```
and student.sno='95005';
```

注: 表和字段的引用方式有两种: 绝对引用和相对引用

- 绝对引用: 数据库名. 表名(. 字段名)
- 相对引用: 表名.(字段名)

数据库基础知识

数据库常用操作

常见报错及解决方法

数据恢复及备份

课程回顾, Q&A

- 1.数组越界
- 2.违反唯一索引
- 3.新版本功能上线，数据库新增表缺失（老表新增加字段缺失等）
- 4.DB权限赋权（开发新建表后无查询、修改、删除某张表数据的权限）
- 5.数据库表死锁
- 6.数据库常见报错编码识别（DB2错误大全）

```
2016-03-15 15:37:52.411 [] [WebContainer : 13] DEBUG
c.suning.oms.res.base.ControllerBase-接口服务端响应报文: <?xml
version="1.0" encoding="UTF-8"?><MbfService><output1><MbfHeader>
<returnFlag>N</returnFlag>
<returnCode>E9999</returnCode>
<returnMessage>保存订单信息出错,
errorMessage=com.suning.framework.exception.BaseException: Code:
SP-DAO-EntityOrderDAOImpl.persist; Message:
SP-DAO-EntityOrderDAOImpl.persist; nested exception is
org.springframework.dao.DataIntegrityViolationException:
PreparedStatementCallback; SQL [INSERT INTO
SO_ORDI_ACTIVITY_INFO(ORDER_ITEM_ID, ACTIVITY_TYPE, ACTIVITY_ID,
SUB_ACTIVITY_TYPE, SEL_ACTIVITY_CODE, CREATED_BY, LAST_UPD_BY)
VALUES (?, ?, ?, ?, ?, ?, ?)]; The value of a host variable in the EXECUTE or
OPEN statement is out of range for its corresponding use.. SQLCODE=-302,
SQLSTATE=22001, DRIVER=3.61.75; nested exception is
com.ibm.db2.jcc.am.SqlDataException: The value of a host variable in the
EXECUTE or OPEN statement is out of range for its corresponding use..
SQLCODE=-302, SQLSTATE=22001, DRIVER=3.61.75</returnMessage>
```

问题排查:

1. 查看对应表插入的字段在对应的表中是否定义
 2. 查看对应表中每个字段的属性及长度定义确定是哪一个字段值超长?
 3. 如果字段值长度违反接口定义长度, 传参不合法, 更改业务数据传入限制
 4. 业务传入值数据长于库里定义长度, 需要扩充数据库表对应字段长度
- Eg. 修改字段类型属性及长度:
- ```
ALTER TABLE
T_SERVIE_ORDER ALTER
COLUMN SERVICE_ID SET
DATA TYPE VARCHAR(12);
```

```
例 : Message: SP-DAO-LogisticStatusDaoImpl.saveLogisticStatus: nested exception is
org.springframework.dao.DuplicateKeyException: PreparedStatementCallback; SQL [INSERT INTO
SO_ORDI_DELI_DETAIL (STATUS_CHARGE_AGAINST_FLAG, ORDER_ITEM_ID, DELI_NUMBER,
DELI_ORDER_ID, JOB_DOCUMENT_TYPE, EXPRESS_ORDER_ID, EXPRESS_COMPANY_NAME, BILL_TYPE,
DISTRIBUTE_CODE, OP_TYPE, STATUS_DESC, STATION_CODE, STATION_DESC, DELI_POINT_NUM,
PLAN_ARRIVEL_TIME, REAL_COMPLETE_TIME, OPERATOR, OPERATOR_NAME, BUS_NO, PHONE_NO, CARREIR,
IS_LAST_STATION, DISMANTLE_ORD_AMOUNT, PACKAGE_ID, SHCEDULING_TIME, RET_CHECK_CODE,
RETURN_STOCK, INSTORE_REASON, ARRANGE_ANOTH_TIME, ARRANGE_REASON, LARGE_SMALL_FLAG,
CREATED_BY, LAST_UPD_BY, ACTIVE_FLAG, REMARK, CURRENT_SITE_PROPERTIES, NEXT_SITE_NAME,
DELIVERYMAN_NO) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,
?, ?)]: One or more values in the INSERT statement,
UPDATE statement, or foreign key update caused by a DELETE statement are not valid because
the primary key, unique constraint or unique index identified by "4" constrains table
"OMSDADM2.SO_ORDI_DELI_DETAIL" from having duplicate values for the index key..
SQLCODE=-803, SQLSTATE=23505, DRIVER=3.61.75; nested exception is
com.ibm.websphere.ce.cm.DuplicateKeyException: One or more values in the INSERT statement,
UPDATE statement, or foreign key update caused by a DELETE statement are not valid because
the primary key, unique constraint or unique index identified by "4" constrains table
"OMSDADM2.SO_ORDI_DELI_DETAIL" from having duplicate values for the index key..
SQLCODE=-803, SQLSTATE=23505, DRIVER=3.61.75
```

注： -803 23505 不能插入行，因为这将违反唯一索引的约束

## 问题排查：

根据错误代码中提示，应该是id有了唯一性约束的问题。最后将当前insert语句放在db2中执行后，仍有此问题，因此断定id字段有问题。由于id字段是自增字段，于是想到的可能性是开发人员之前在数据库中自定义的给了id固定值，导致默认值和此固定值之间冲突。

解决方法样例：alter table student alter ID restart with xxxx 。xxxx可以自定义，为当前表中id的最大值加1即可

- 例：2016-05-16 16:01:10.293 [] [pool-10-thread-2] ERROR  
com\_suning\_exception\_log-com.suning.framework.exception.BaseException  
been intercepted:↵  
Code: SP-DAO-EntityOrderDAOImpl.persist^MMessage: SP-DAO-EntityOrderDAOImpl.persist;  
nested exception is org.springframework.jdbc.BadSqlGrammarException:  
PreparedStatementCallback; bad SQL grammar [INSERT INTO SO\_ORD(ORDER\_ID,  
B2C\_ORDER\_ID, ORDER\_TIME, ORDER\_RES, ORDER\_CHANNEL, ORDER\_COMMENTS,  
PAY\_ITEM\_NUM, ORDER\_CATEGORY, VERI\_CODE, REAL\_PAY\_AMOUNT, ORDER\_STATUS,  
ORDER\_SALE\_AMOUNT, ORDER\_PAY\_AMOUNT, USER\_ID, TRANSPORT\_FEE, IS\_INVOICE,  
SERVICE\_AMOUNT, TOTAL\_ITEM, KZ\_GROUP\_CARD, BILL\_TYPE, CLIENT\_ID, MEM\_LEVEL,  
MEM\_NAME, MEM\_TYPE, MEM\_OUT\_CARD\_NO, MEM\_IN\_CARD\_NO, CREATED\_BY,  
LAST\_UPD\_BY, CREATED\_TIME, REMARK, MEM\_CARD\_ID, MEN\_DBNO,  
RECEIVE\_ORDER\_TYPE, ECO\_TYPE, ORG\_ID, COST\_CENTER\_CODE, BALNCE\_TYPE,  
CART\_TWO\_NO, ORDER\_PAY\_TYPE, PAY\_OP\_TYPE ) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?,  
?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ? ); nested exception is  
com.ibm.db2.jcc.am.SqlSyntaxErrorException: "ORDER\_PAY\_TYPE" is not valid in the context  
where it is used.. **SQLCODE=-206, SQLSTATE=42703, DRIVER=3.61.75**↵  
注： -206 42703 列名没有在任何表中，或者没有在  
定义触发器所在的表中 ↵

由于业务扩展需要，在原有表中需增加一个或者多个字段，在发布应用代码前，需要将对应数据库表中对应新加字段加上，不然业务触发插入或者更新对应表字段数据的时候会报上述错误。

SQL语句样例：ALTER  
TABLE OMSADM. SO\_ORD  
ADD COLUMN  
ORDER\_PAY\_TYPE  
STRING(4);



```
20 UPDATE IDX_OMS2DBREDIS_EXETIME SET IDX_OMS2DBREDIS_EXETIME.EXE_UPDATE_TIME='2016-06-02 18:10:00';

13:74 [629] INS Auto Commit: ON Unitt
Log to GUI Log to File
... Physical database connection acquired for: PRE-OMS-omsprepub
20:11:29 [UPDATE - 0 row(s), 0.000 secs] [Error Code: -551, SQL State: 42501] DB2 SQL Error: SQLCODE=-551, SQLSTATE=42501,
SQLERRMC=YUNWEI;UPDATE;OMSADMOL.IDX_OMS2DBREDIS_EXETIME, DRIVER=3.57.82
... 1 statement(s) executed, 0 row(s) affected, exec/fetch time: 0.000/0.000 sec [0 successful, 0 warnings, 1 errors]
```

注：-551 42501 指用户试图对不拥有权限的特定的对象进行操作，或者表不存在  
新增表未赋予对应的执行操作权限：

赋权语句样例：

```
grant update,select,insert on table omsadm.OMS_DELIVERY_RULE TO USER yunwei;
grant update omsadm.MD_DELIVERY_REGION_LOCATION TO USER yunwei;
```

报错样例：

(DB2 的数据库)在做 update 更新的时候,发生了死锁。后台报的错误为:SQLCODE=-911, SQLSTATE=40001

SQLCODE=-911, SQLSTATE=40001 错误的原因:是在执行 update 语句的时候发生了死锁

解决办法:

db2 命令行,

1、用管理员用户登录: db2 connect to 你的数据库名 user 用户名 using 密码

2、db2 "get snapshot for locks on 数据库名"

————上面语句执行完成以后,你可以找到下面一段文字

|              |                           |
|--------------|---------------------------|
| 应用程序句柄       | = 689                     |
| 应用程序标识       | = *LOCAL.DB2.120711101108 |
| 序号           | = 00001                   |
| 应用程序名        | = javaw.exe               |
| CONNECT 授权标识 | = DB2ADMIN                |
| 应用程序状态       | = UOW 正在等待                |

3、db2 "force application(689)" 689就是上面查询出来的应用程序句柄

杀掉死锁进程

常见报错: **SQLCODEXXX=XX, SQLSTATE=XX**

-638 42601 在CREATE TABLE语句中缺少列定义

-644 42615 在SQL语句中为关键词指定的值无效

-910 57007 因为在该对象上挂起DROP, 所以不能访问该对象

-913 57033 因为死锁或超时导致不成功执行

-918 51021 SQL语句不能被执行, 因为连接丢失

-923 57015 因为DB2不可用, 所以不能建立连接

-206 42703 列名没有在FROM语句所引用的任何表中, 或者没有在定义触发器所在的表中

-803 23505 不能插入行, 因为这将违反唯一索引的约束



DB2错误大全

数据库基础知识

数据库常用操作

常见报错及解决方法

数据恢复及备份

课程回顾, Q&A

## 数据备份几大分类：

### 1.完全备份

这是大多数人常用的方式，它可以备份整个数据库，包含用户表、系统表、索引、视图和存储过程等所有数据库对象。

但它需要花费更多的时间和空间，所以，一般推荐一周做一次完全备份。

### 2.事务日志备份

事务日志是一个单独的文件，它记录数据库的改变，备份的时候只需要复制自上次备份以来对数据库所做的改变，

所以只需要很少的时间。为了使数据库具有鲁棒性，推荐每小时甚至更频繁的备份事务日

### 3.差异备份

它是只备份数据库一部分的另一种方法，它不使用事务日志，

相反，它使用整个数据库的一种新映像。它比最初的完全备份小，因为它只包含自上次完全备份以来所改变的数据库。它的优点是存储和恢复速度快。

推荐每天做一次差异备份。

### 4.文件备份

数据库可以由硬盘上的许多文件构成。如果这个数据库非常大，并且一个晚上也不能将它备份完，那么可以使用文件备份每晚备份数据库的一部分。

由于一般情况下数据库不会大到必须使用多个文件存储，所以这种备份不是很常用。

按照数据库的状态可分为三种：

- 1.冷备份，此时数据库处于关闭状态，能够较好的保证数据库的完整性。
- 2.热备份，数据库正处于运行状态，这种方法依赖于数据库的日志文件进行备份。
- 3.逻辑备份，使用软件从数据库中提取数据并将结果写到一个文件上。

## 1.如何执行DB2命令

在Linux中切换用户到db2inst1

```
#su – db2inst1
```

然后可以执行命令

1、直接在db2inst1用户环境下运行的，比如：db2stop、db2start

2、需要 db2 + 空格 + 命令 这种方式运行

这种命令可以通过以下两种方式执行

A、在命令行方式下执行“db2 命令”

B、先执行“db2”命令，然后在db2的提示符下，直接键入命令执行

连接数据库命令

```
db2 => connect to OA
```

Database Connection Information

Database server = DB2/LINUX 8.2.2

SQL authorization ID = DB2INST1

Local database alias = OA

1. 离线备份(必须在数据库所在PC机进行操作)
2. 在线备份
  - 注意：如果要做在线的备份，要对两个参数作修改：USEREXIT和LOGRETAIN，将其修改为on，默认是off的，默认使用日志模式，修改完参数后，可以变为归档日志模式，启用了userexit做数据库日志的归档。
3. 增量备份
  - 注意：必须要修改trackmod为on状态，并有一次完全备份。



**DB2数据备份与恢复**

```
db2move dbname export
```

```
#su – db2inst1
```

```
[db2inst1@test34]$mkdir back
```

因为dbmove备份到当前目录下，建议先建立个目录用来存放备份文件

```
[db2inst1@test34]$cd back
```

```
[db2inst1@test34 back]$db2move test export
```

```
***** DB2MOVE *****
```

```
Action: EXPORT
```

```
Start time: Thu Dec 21 18:28:30 2015
```

```
Connecting to database TEST ... successful! Server: DB2 Common Server V8.2.2
```

```
Disconnecting from database ... successful!
```

```
End time: Thu Dec 21 18:28:30 2015
```



```
db2move dbname import
```

数据库的还原

```
#su – db2inst1
```

```
db2inst1 @test34]$cd back //进入还原备份目录
```

```
[db2inst1 @test34 back]$db2move test1 import
```

//把test数据库还原到test1数据库中，test1数据库必须实际存在。

```
***** DB2MOVE *****
```

```
Action: IMPORT
```

```
Start time: Thu Dec 21 18:30:03 2015
```

```
Connecting to database TEST1 ... successful! Server: DB2 Common Server V8.2.2
```

```
Disconnecting from database ... successful!
```

```
End time: Thu Dec 21 18:30:05 2015
```

- **数据库基础知识**
  1. 数据、信息、数据处理
  2. 数据库系统结构及特征介绍
  3. 数据模型介绍
  
- **数据库常用操作**
  1. SQL主要语句语法
  2. 表级完整性约束条件
  3. 数据定义语句语法介绍（DDL）
  4. 数据操作语句语法介绍（DML）
  
- **测试常见数据库报错样例及解决办法**
  1. 数组越界
  2. 违法唯一索引
  3. 表操作权限赋权等
  
- **数据的恢复及备份**

数据备份的种类、备份方式

# 总结提问:



关系型数据模型中的“元组”和“属性”分别代表什么？

二维表中的“行”即关系数据模型中的“元组”，  
二维表中的“列”即关系数据模型中的“属性”



用户是用DML还是DDL完成下列任务？

- a. 改变客户地址
- b. 定义目录表
- c. 给某个数据库用户增加修改数据权限

- a. DML（插入、查找、修改、删除等操作）
- b. DDL（数据定义功能）
- c. DCL（数据库的运行和管理）



多表查询常见有哪些方法？

- 1.
- 2.
- 3.

THANK YOU  
谢 谢

SUNING 苏宁

