

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、信息、数据处理

计算机数据管理技术的发展

数据库、数据库管理系统、数据库系统

数据库的结构及特征


数据模型介绍

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？
信息=数据+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，数据库中可以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，完全依靠于程序	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

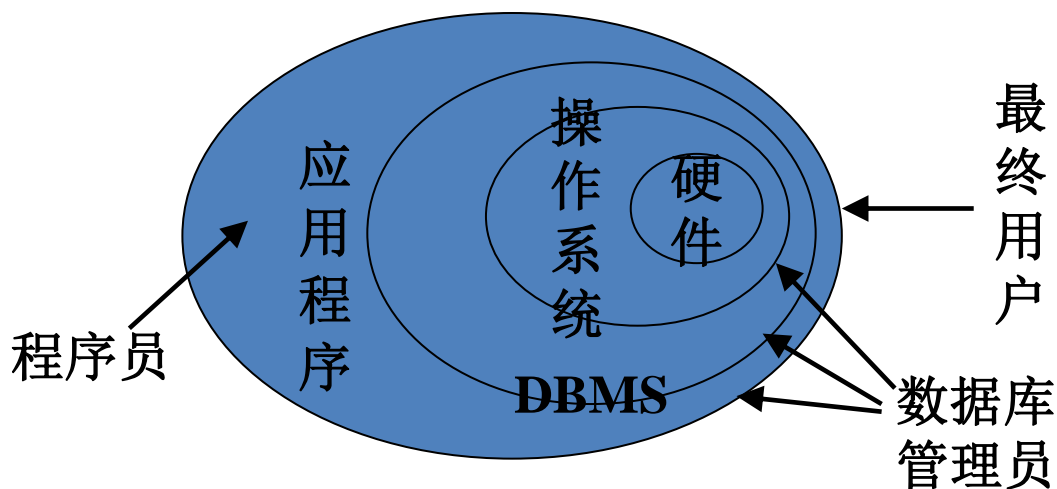
(1) 数据库 (DataBase, 简称 DB)

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。

注：常将数据库系统简称为数据库。

a、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。

①**数据定义功能**：数据定义语言（Data Definition Language, 简称DDL）

②**数据操纵功能**：数据操纵语言（Data Manipulation Language, 简称DML）

如：插入、查找、修改和删除等操作。

③**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

④**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）

<1>数据安全性（Security）保护。

<2>数据完整性（Integrity）检查。

完整性：数据的正确性、有效性、一致性。

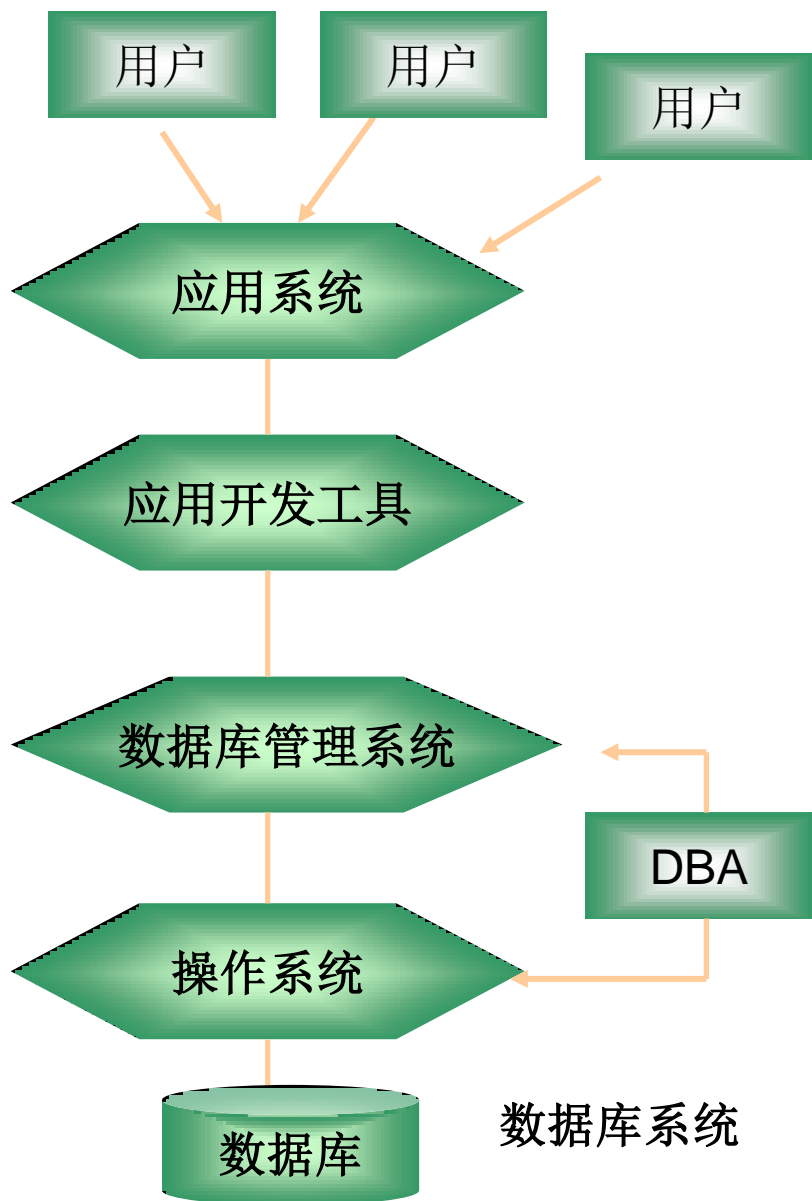
<3>并发（Concurrency）控制：能实现多个用户同时对数据库库中的数据或同一数据进行合理存取。

<4>数据库恢复（Recovery）。

注：目前典型的关系数据库管理系统（RDBMS）有

Oracle 、 Sybase、 DB2、 Informix、 SQL server、 Foxpro、 access。

- Oracle
 - 应用广泛、功能强大，分布式数据库系统
 - “关系-对象”型数据库
- MySQL
 - 快捷、可靠
 - 开源、免费、与PHP组成经典的LAMP组合
- SQL Server
 - 针对不同用户群体的五个特殊的版本
 - 易用性好
- DB2
 - 应用于大型应用系统，具有较好的可伸缩性



数据库系统的主要特征：

- ①实现数据共享
- ②减少数据冗余度
- ③保持数据的一致性
- ④数据的独立性
- ⑤安全保密性
- ⑥并发控制
- ⑦故障恢复

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...）

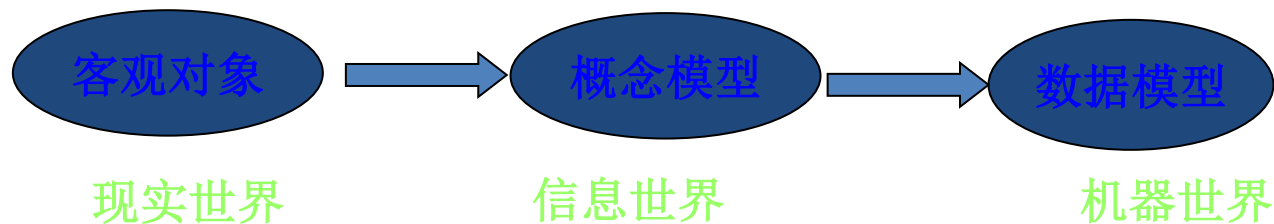
注意：模式中不包含具体的记录数据

模式的实例：模式的具体值称为模式的实例。

数据库模式和实例
的关系？

- 模式与实例的关系：
 - 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。
- **数据库中的模式：**用数据描述语言精确定义数据模型的全部语句。

- 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）

系统A

系统B

系统C

...

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）

某网站数据库

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、用户基本信息表中增加一个字段，对3个系统有影响吗？
2、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

SQL主要语句语法

表级完整性约束条件

数据操作语句

CREATE语法

- CREATE DATABASE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

RENAME TABLE语法

- RENAME TABLE语法

- 1. CREATE DATABASE语法
- CREATE DATABASE用于创建数据库，并进行命名。如果要使用CREATE DATABASE，您需要获得数据库CREATE权限。

CREATE DATABASE *db_name* ;

例如：Create Database test1;

- CREATE TABLE**<表名>
(<列名><数据类型> [列级完整性约束条件]
[,<列名><数据类型> [列级完整性约束条件]]...
[,<表级完整性约束条件>]);

注：列级约束条件有两个任选项：

NOT NULL，表示此列不得置NULL，在其后还可加**UNIQUE**任选项，表示列值不得重复。

DEFAULT，当此列的值空缺时，填以缺省值

例如：stu_id char(20) not null,
price float default 0.01

```
CREATE TABLE
omsadm.MD_DEF_INV
(
    CATALOG_CODE VARCHAR(12)
NOT NULL,
    SWITCH_FLAG VARCHAR(12),
    EXTENSION VARCHAR(12),
    VERSION_NO VARCHAR(12),
    CONSTRAINT P_PRIMARYKEY
PRIMARY KEY (CATALOG_CODE)
) IN "HOT_SPACE" INDEX IN
"IDX_HOT_SPACE";
```

```
COMMENT ON TABLE
omsadm.MD_DEF_INV
IS
    '默认发票目录表';
```


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	

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

×

0010014	雷铜	江西南昌
---------	----	------

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

- 常见完整性约束：
 - PRIMARY KEY 主码约束(主键)
 - 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）
 - CHARACTER SET name 指定字符集（仅适用字符串）

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，MySQL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。
- 举例：
 - `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。
- 举例：
 - `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

- 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

- 定义外关键字约束的语法如下：

```
CONSTRAINT FOREIGN KEY (column_name1[, column_name2, ..., column_name16])  
REFERENCES ref_table [ (ref_column1[, ref_column2, ..., ref_column16] ) ]  
[ ON DELETE { CASCADE | NO ACTION } ]  
[ ON UPDATE { CASCADE | NO ACTION } ]
```

- 各参数说明如下：

REFERENCES

指定要建立关联的表的信息。

ref_table

指定要建立关联的表的名称。

ref_column

指定要建立关联的表中的相关列的名称。

- 例：create table orders(
order_id char(8),
p_id char(8),
p_name char(10) ,
constraint primary key (order_id) ,
foreign key(p_id) references products(p_id)
on delete restrict) ;

- ON DELETE {NO ACTION|RESTRICT}
指定在删除表中数据时，对关联表所做的相关操作。在子表中有数据行与父表中的对应数据行相关联的情况下，
NO ACTION 是缺省值。
RESTRICT：凡被基表引用的主键，不得删除
- ON UPDATE {NO ACTION|RESTRICT}
指定在更新表中数据时，对关联表所做的相关操作。在子表中有数据行与父表中的对应数据行相关联的情况下，
NO ACTION 是缺省值。
RESTRICT：凡被基表引用的主键，不得删除

- CREATE INDEX *index_name* ON *tbl_name* (*index_col_name*,...)

注: *index_col_name*: *col_name* [(*length*)] [ASC | DESC]

例如: create index stu_age on student(age ASC);

- DROP DATABASE *db_name*

删除已有的数据库.

- ALTER TABLE语法

- 例:

- Alter table aa add primary key(id);
- Alter table aa add name char(20);
- Alter table aa drop primary key;
- Alter table aa modify id char(100) not null;

- ALTER TABLE *tbl_name*

[ADD<新列名><数据类型>[完整性约束]]

[DROP <列名><完整性约束名>]

[MODIFY<列名><数据类型>];

- ADD用于增加新列和新的完整性约束条件, 新增加的列一律为空值。(index, primary key等)
- DROP用于删除指定的完整性约束条件 (index,primary key, foreign key)
- MODIFY用于修改原有的列定义

- CREATE INDEX *index_name* ON *tbl_name* (*index_col_name*,...)

注: *index_col_name*: *col_name* [(*length*)] [ASC | DESC]

例如: create index stu_age on student(age ASC);

- DROP DATABASE *db_name*

删除已有的数据库.

- ALTER TABLE语法

- 例:

- Alter table aa add primary key(id);
- Alter table aa add name char(20);
- Alter table aa drop primary key;
- Alter table aa modify id char(100) not null;

- ALTER TABLE *tbl_name*

[ADD<新列名><数据类型>[完整性约束]]

[DROP <列名><完整性约束名>]

[MODIFY<列名><数据类型>];

- ADD用于增加新列和新的完整性约束条件, 新增加的列一律为空值。(index, primary key等)
- DROP用于删除指定的完整性约束条件 (index,primary key, foreign key)
- MODIFY用于修改原有的列定义

RENAME TABLE语法

- RENAME TABLE *tbl_name* TO *new_tbl_name*

对一个表重命名

例如: `rename table student to ss;`

创建，删除视图(view)

- 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>
```

例如:

```
Create view stu_test(id,name)
As select stu_id,name from student;
```

- Drop view stu_test;

- Select
- Update
- Insert
- Delete

Select查询语句

- **SELECT** [ALL|DISTINCT] <目标列表达式>
[,<目标列表达式>]...
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]];

- **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，该属性列值相等的元组为一组
- **ORDER**将结果表按列名2的值升序或降序排序

Where

查询满足条件的元组（**where** 子句）

1. 比较大小 取出某特定范围内数据
2. 确定范围（**BETWEEN...AND/Not Between...and**）
3. 确定集合（**IN, NOT IN**）
4. 字符匹配（**LIKE , NOT LIKE**）

可适用通配符 %和 _。

- %:代表任意长度（长度可以为0）的字符串:a%
- _: 代表任意单个字符: a_

例如:

- Select * from student;
- Select * from student where age not between 18 and 19;
- Select * from student where name like a%;
- Select student.name, teacher.name from student,teacher where student.select_course_id=teacher.teach_course_id;

- 一般格式:

UPDATE <表名>

SET<列名>=<表达式>[,<列名>=<表达式>]...

[WHERE<条件>];

- 功能是修改指定表中满足条件的元组
- SET子句给出<表达式> 的值用于取代相应的属性列值。
- 省略WHERE 子句，表示修改表中所有元组

例如:

Update set address='asdf',birthday='1977-11-3' where stu_id=2;

- 一般格式:

```
INSERT INTO <表名>[(<属性列1>[,<属性列2>...])  
VALUES(<常量1>[,<常量2>]...)
```

- 功能是将新元组插入指定的表中
- INTO中没有出现的属性列，新纪录在其上取空值
- 属性列省略则按已有关系模式顺序插入，且其在每个属性列上均有值。

```
Insert into student values ('1','aa','asdf','1980-12-09');
```

```
Insert into student(stu_id,name) values('2','bb');
```


DELETE FROM<表名> [WHERE<条件>];

- 功能是从指定表中删除满足条件的所有元组
- 如果省略WHERE子句，表时删除表中全部元组，但表的定义仍在数据字典中。即：只删除数据。

例如：

Delete from student where name='aa';

- 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个个字段值超长?
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

业务传入值数据长度较之原有长度, 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);
```

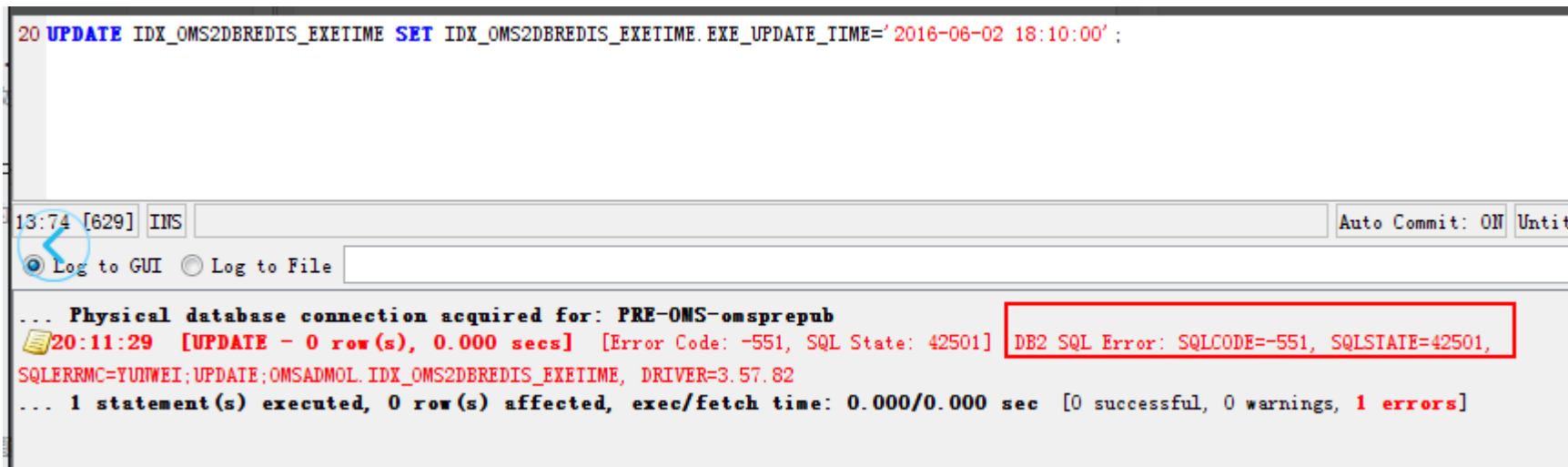

- 解决方法样例：`alter table student alter ID restart with xxxx`。`xxxx`可以自定义，为当前表中id的最大值加1即可

- 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

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** VARCHAR(10);

报错样例：



```
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]
```

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** 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

数据备份的种类和不同？

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象。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。



DB2数据备份与恢复

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。


```
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
```


1.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”。

2.用户是用DML还是DDL完成下列任务？

a.改变客户地址

a. DML（插入、查找、修改、删除等操作）

b.定义目录表

b. DDL（数据定义功能）

c.给某个数据库用户增加修改数据权限

c. DCL (数据库的运行和管理)

3.目前典型的关系数据库管理系统（RDBMS）有哪些？

Oracle、Sybase、DB2、Informix、SQL server、Foxpro、access。

THANK YOU
谢 谢

SUNING 苏宁

