数据库： SQL - 第一部分

课程和练习

SQL：一个标准

- 向后兼容
- ANSI/ISO
  - SQL-86 - IBM SQL实现的交叉点
  - SQL-89 - 小型修订，完整性约束
  - SQL-92 - 模式修改，事务，集合运算符，新的数据类型，游标，参考完整性动作，...
  - SQL:1999 - 递归查询、触发器、对象关系特征、正则表达式、全文类型、图像、空间数据等。
  - SQL:2003 - SQL/XML，序列生成器
  - SQL:2006 - XML的其他扩展，XQuery的整合
  - SQL:2008
  - SQL:2011 - 时间性数据库

isep
École d'ingénieurs du numérique

SQL语句类别

- 数据库要素的**定义**
  - *数据定义语言*，或称DDL
- 数据处理
  - *数据操作语言*，或称DML
- 数据**访问权限的管理**
  - *数据控制语言*，即DCL
- **交易管理**
  - *交易控制语言*，即TCL
- **集成式SQL**

- *嵌入式SQL*

SQL： 数据定义语言

- **创建**
- **ALTER**
- **撤消**
- 以及其他指示：
  - 审计、无审计、分析、重命名、截断

SQL： 数据操作语言

- **插入**
- **更新**
- **DELETE**
- **选择**
- 以及其他指示：
  - 解释，计划，锁定表

数据定义语言

创建、更改、删除

isep
École d'ingénieurs du numérique

CREATE：创建一个数据库

https://dev.mysql.com/doc/refman/8.0/en/charset-database.html

CREATE DATABASE [IF NOT EXISTS] database_name

[CHARACTER **SET** charset_name]。

[COLLATE collation_name]

isep
École d'ingénieurs du numérique

字符集和排序

http://www.dynamic-mess.com/sql/comprendre-charset-et-collation/

- **字符集**： 一组符号和编码
- **互相排列（整理）**： 一套比较字符集中的字符的规则
- 一个游戏可以有几个等级间，通常每个语言有一个等级。
  - 这允许例如按字母顺序对字符进行排序=>DBMS将能够解释排序查询，但也能知道一个字符是否与另一个字符等价，例如'*a*'和'*to*'。

键入以下SQL命令以找出Mysql中的所有字符集及其默认顺序： **SHOW CHARACTER SET;**

CREATE：分类间的后缀

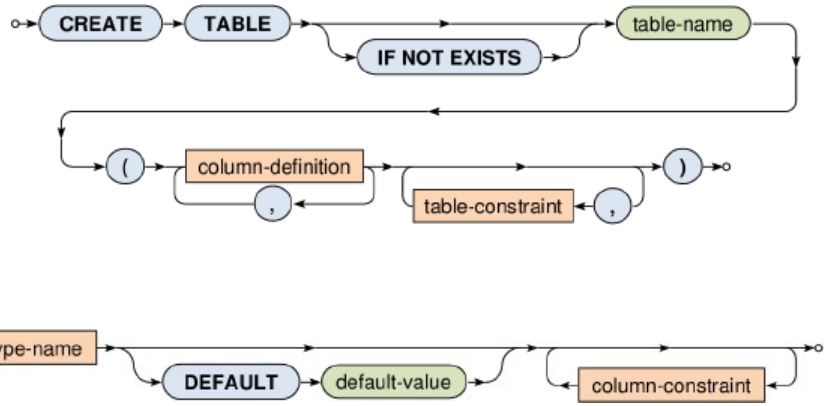
ÄÄÄ	不敏感的口音
作为	重音敏感
这里	不区分大小写
ÄÄÄ	区分大小写
许可证	加纳敏感
滨	二进制

- latin1_general_ci明确地不区分大小写，隐含地不区分重音。
- latin1_general_cs明确区分大小写，隐含区分重音。
- utf8mb4_0900_ai_ci是明确不区分大小写的。

CREATE：简单地创建一个表和相关栏目

- 建立一个表格
  - 表名
  - 每一栏的定义
    - 列的名称
    - 列的数据类型
    - 默认值

CREATE TABLE table table_name **(**
column1 data_type, column2 data_type, column3 data_type, column4 data_type **)**



CREATE：数据类型

- **INTEGER**：该类型用于存储在4个字节上编码的有符号整数。
- **VARCHAR(length)**：该数据类型用于存储可变长度的字符串。长度必须小于2000，没有默认值。
- **DATE**: 该数据类型允许存储由日期组成的数据。
- **TIMESTAMP**：该数据类型允许存储由日期和时间组成的数据。
- **BOOLEAN**: 该数据类型允许存储布尔值
- **CHAR(length)**：该数据类型用于存储固定长度的字符串。长度必须小于255，其默认值为1

-

CREATE：数据类型 (续)

- BIGINT：该类型用于存储8字节编码的有符号整数。
- REAL：该类型允许存储6位有效数字的实数，编码在4个字节上。
- 双精度：这种类型允许在8个字节上存储有15个有效数字的实数编码。
- NUMERIC[(precision, [length])]：这种数据类型可以存储整数和实数数据，精度为1000有效数字。
  - length指定存储的最大有效数字数，precision给出小数点后的最大数字数。
- MONEY: 这种数据类型用于存储货币价值。
- TEXT：该数据类型用于存储长度可变的字符串。

创建：无约束的例子

产品(id, name, price, produced, available, weight)

```
CREATE TABLE Product (
        id INTEGER、
        name VARCHAR(128)、
        价格 DECIMAL(6,2)、
```

```
    生产日期、
    available BOOLEAN DEFAULT TRUE、
    重量 FLOAT
);
```

CREATE: 创建时有完整性约束

- 建立一个表格
  - 表名
  - 每一栏的定义
    - 列的名称
    - 列的数据类型
    - 默认值
    - **列的完整性约束**
  - **表的完整性约束**

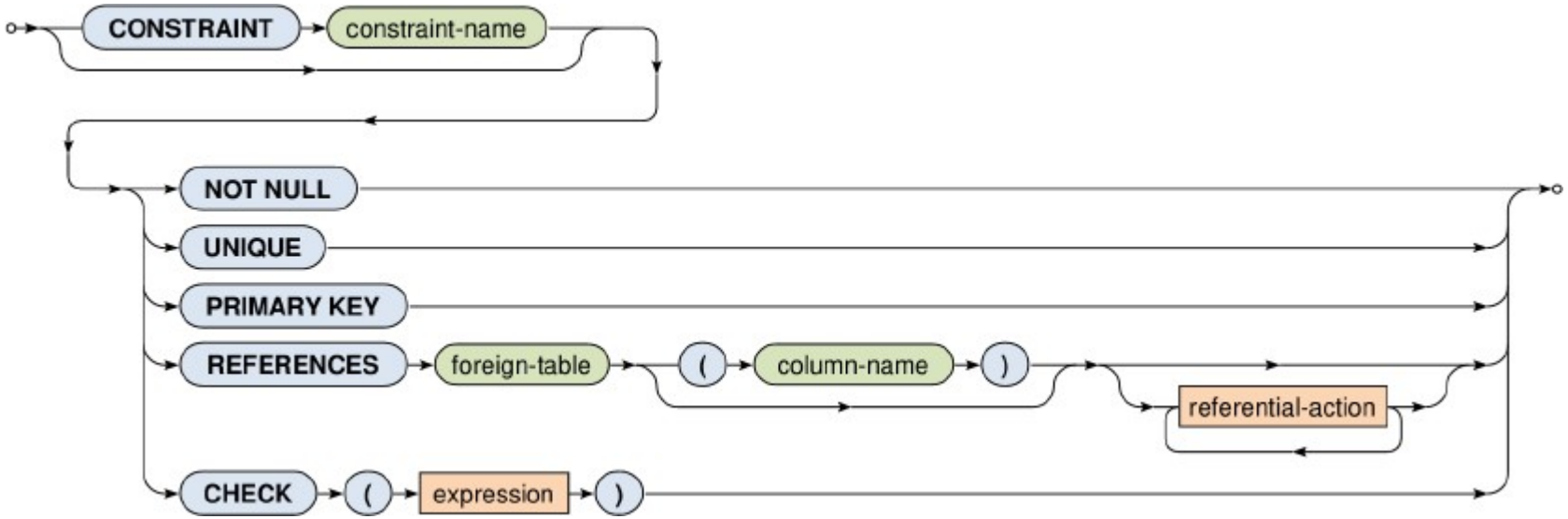
CREATE: 列的完整性约束

不是空的	防止一个列的空值被记录下来
NULL	允许为一个列记录一个空值
獨特的	指定该属性为表的次要键。两个图元不能收到相同的该列的值，但允许插入NULL值
DEFAULT值	在向表中添加行时，如果没有为该列指定数据，则指定一个默认值
PRIMARY KEY	指定该属性为表的主键。相当于UNIQUE NOT NULL约束。

CREATE: 列的完整性约束（续）

FOREIGN KEY [列] REFERENCES表[(列)] *[关于删除级联]*	被定义的表的属性的参考完整性约束。这个属性的值必须存在于外域表'table'的列中，这个列有一个PRIMARY KEY或UNIQUE约束。 如果没有指定列属性，则使用与'表'的主键对应的属性。
检查（条件）	在插入图元时检查该属性是否满足条件条件。

CREATE: 列的完整性约束 (摘要)



CREATE：列的完整性约束（示例）

Person(id, personalNumber, address, age, serialNumber, color)

```
CREATE TABLE Person(
        id INTEGER PRIMARY KEY、
        personalNumber INTEGER,
        address VARCHAR(256),
        age INTEGER、
        serialNumber INTEGER NOT NULL、
        color STRING
);
```

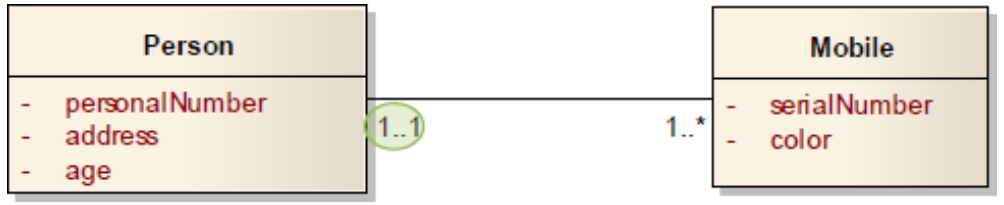
CREATE：列的完整性约束（示例）

```
CREATE TABLE Producer (
   id INTEGER PRIMARY
   KEY, name VARCHAR(128)
   、
   country VARCHAR(64)
);
```

命名一个约束，在**列**约束的情况下**不是强制性的**

```
CREATE TABLE Product (
   id INTEGER CONSTRAINT IC_Product_PK PRIMARY KEY,
   name VARCHAR(128) UNIQUE、
   price DECIMAL(6,2) CONSTRAINT IC_Product_Price NOT NULL,
   produced DATE CHECK (produced >= '2015-01-01'),
   available BOOLEAN DEFAULT TRUE NOT NULL、
   weight FLOAT,
   producer INTEGER、
   FOREIGN KEY(producer) REFERENCES Producer(id)
);
```

CREATE：列的完整性约束（示例）




```
CREATE TABLE Person(
   personalNumber INTEGER PRIMARY KEY,
   address VARCHAR(256)、
   年龄INTEGER、
);
CREATE TABLE Mobile(
   serialNumber INTEGER PRIMARY KEY,
   color STRING、
   personalNumber INTEGER、
   FOREIGN KEY(personalNumber) REFERENCES Person(personalNumber)
);
```

Person(personNumber, address, age) **Mobile**(serialNumber, color, personalNumber) Mobile.personNumber⊆ Person.personNumber

CREATE: 表的完整性约束

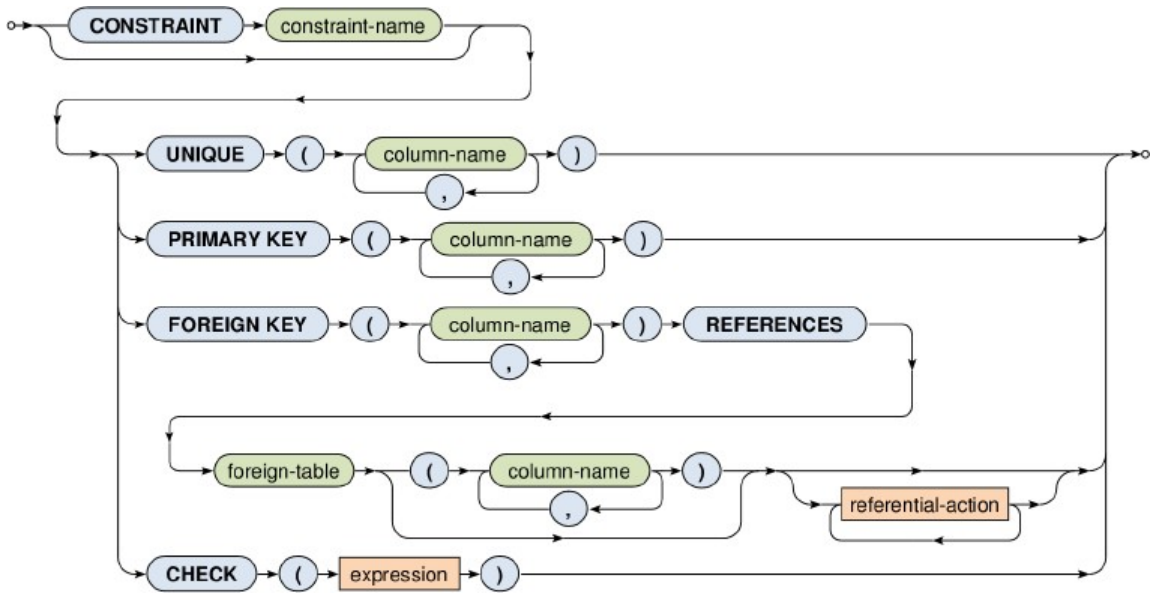
类似于列的约束，但对多列而言

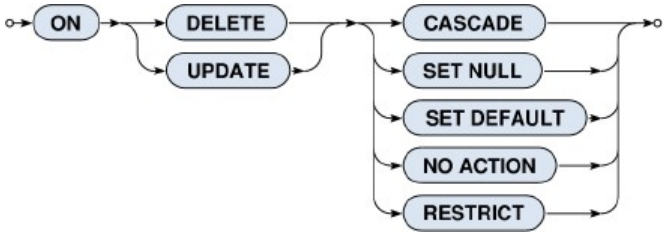
不是空的	防止一个列的空值被记录下来
钮扣	允许为一个列记录一个空值
獨特的	指定该属性为表的次要键。两个图元不能收到相同的该列的值，但允许插入NULL值
DEFAULT值	如果在向表中添加行时没有为该列指定数据，则指定一个默认值
PRIMARY KEY	指定该属性为表的主键。相当于UNIQUE NOT NULL约束。

CREATE: 表的完整性约束（续）

FOREIGN KEY (column...) REFERENCES table [（ column）]* *[关于删除级联]*	被定义的表的属性的参考完整性约束。这个属性所取的值必须存在于表的PRIMARY KEY或UNIQUE约束的列属性中。*表'中具有PRIMARY KEY或UNIQUE约束的列属性中*。如果没有指定列属性，使用的属性是与表的主键相对应的属性。 指定的表格。
检查 （条件）	在插入图元时检查该属性是否满足条件条件。

CREATE：表的完整性约束（摘要）





isep
École d'ingénieurs du numérique

CREATE：表的完整性约束（示例）

生产者(姓名, 国家)

产品(id,., producerName, producerCountry) 产品(producerName, producercountry)⊆生产者(名称, 国家)

```
CREATE TABLE Producer (
  name VARCHAR(128)、
  country VARCHAR(3)、
  CONSTRAINT IC_Producer_PKPRIMARY KEY (name,
       country)
);
```

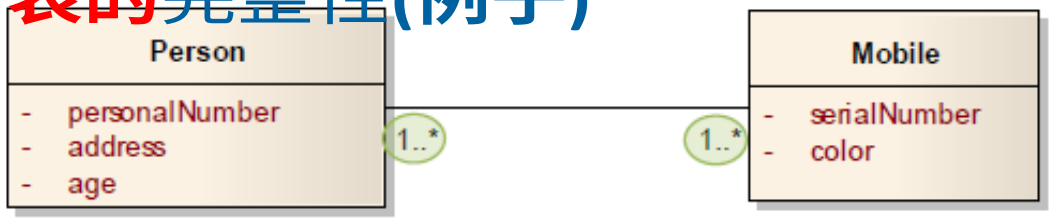
```
CREATE TABLE Product (
  id INTEGER PRIMARY KEY、
  ...
  producerName VARCHAR(128),
  producerCountry VARCHAR(3)、
  CONSTRAINT IC_Product_Producer_FK
             FOREIGNKEY
(producerName、
      producerCountry)
```

参考资料 生产者（姓名，国家）

```
);
```

CREATE：限制条件

表的完整性(例子)



Person(personNumber, address, age)
Mobile(serialNumber, color)
Ownership(personalNumber, serialNumber)
所有权.个人编号⊆个人.个人编号 所有权.序列号⊆手机.序列号

```
CREATE TABLE Person(
  personalNumber INTEGER PRIMARY KEY,
  address VARCHAR(256)、
  年龄INTEGER、
);
CREATE TABLE Mobile(
  serialNumber INTEGER PRIMARY KEY,
  color STRING、
);
```

```
CREATE TABLE Ownership(
  personalNumber INTEGER NOT NULL)
    REFERENCES Person(personalNumber),
  serialNumber INTEGER NOT NULL
    REFERENCES Mobile(serialNumber)、
  限制初级
KEY(personalNumber,serialNumber)
);
```

CREATE：参考性完整性

- **参考完整性**是**对数据库设计者的**一种**自我约束，以确保**存储**数据的完整性。**
- 检查一个外键值是否作为另一个表的主键值存在。
- 例子：发票被链接到一个客户。如果发票已经存储在数据库中，参考完整性将防止客户被删除。另外，如果一个客户是删除，那么它的所有发票也会被删除

CREATE：参考性完整性（示例）

```
CREATE TABLE Producer (
   id INTEGER PRIMARY KEY、
   name VARCHAR(128)、
   country VARCHAR(64)
);

CREATE TABLE Product (
   id INTEGER PRIMARY KEY、
   ...
```

如果一个生产者被删除，那么产品表中所有出现这个生产者的行（图元）都会被删除。

```
    生产者 INTEGER
      REFERENCES Producer (id) ON DELETE CASCADE
);
```

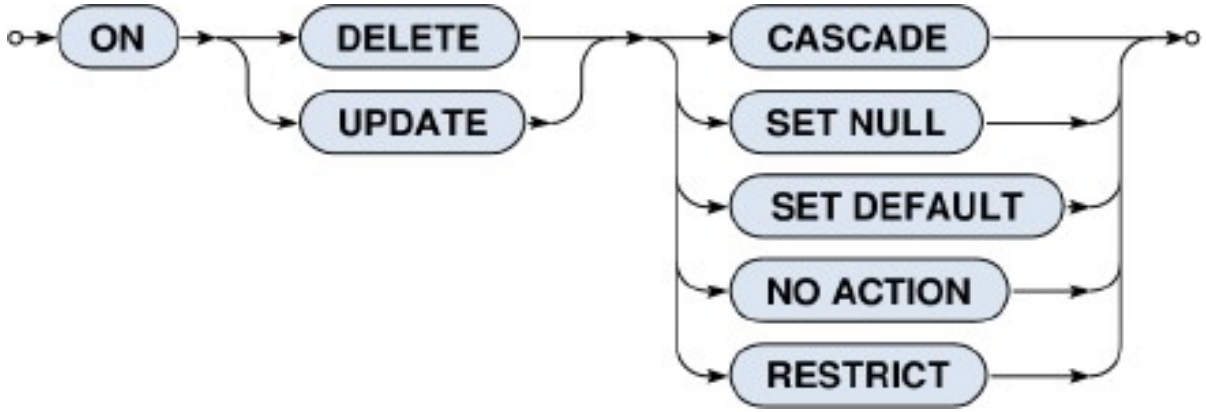
CREATE：参考性完整性

- 触发的情况
  - **更新时、删除时**
    - 当行动被触发时
    - 再一次，这些被认为是对引用表的操作

- 参照行动
  - **卡斯卡德**
    - 具有引用值的行也被更新/删除
  - **SET NULL** - 引用值被设置为NULL
  - **SET DEFAULT** - 引用值被设置为其默认值
  - **无行动**--默认--无行动发生

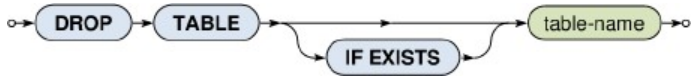
- 也就是说，就像根本不会定义任何指称性动作一样

CREATE：参考性完整性（摘要）



撤消

- 对表格创建的补充
  - 也就是说，表的定义和表的内容都被删除。



ALTER

- 增加/改变/删除表的列/完整性约束