数据库系统概论

ch1：

概念类：

数据库系统DBS

包含DB、DBMS、开发工具、应用程序和数据库管理员， 存储、管理和处理数据的信息系统。

数据库管理系统DBMS

获取和维护数据的系统软件，位于用户与操作系统之间 的数据管理软件。

数据库DB

长期储存在计算机内、有组织的、可共享的大量 数据的集合。

数据Data

描述事物的符号记录

数据模型

用来抽象、表示和处理现实世界数据和 信息的一种方法

概念模型、逻辑模型、物理模型

概念模型：

实体：概念模型的单位（一个学生，一门课，一件商品）

属性：实体所具有的特征

码：**唯一**标识一个实体的属性集

逻辑模型：

按计算机观点对数据和信息进行建模（和语义相关）

层次模型　　网状模型　　关系模型　　面向对象模型　　关系对象模型

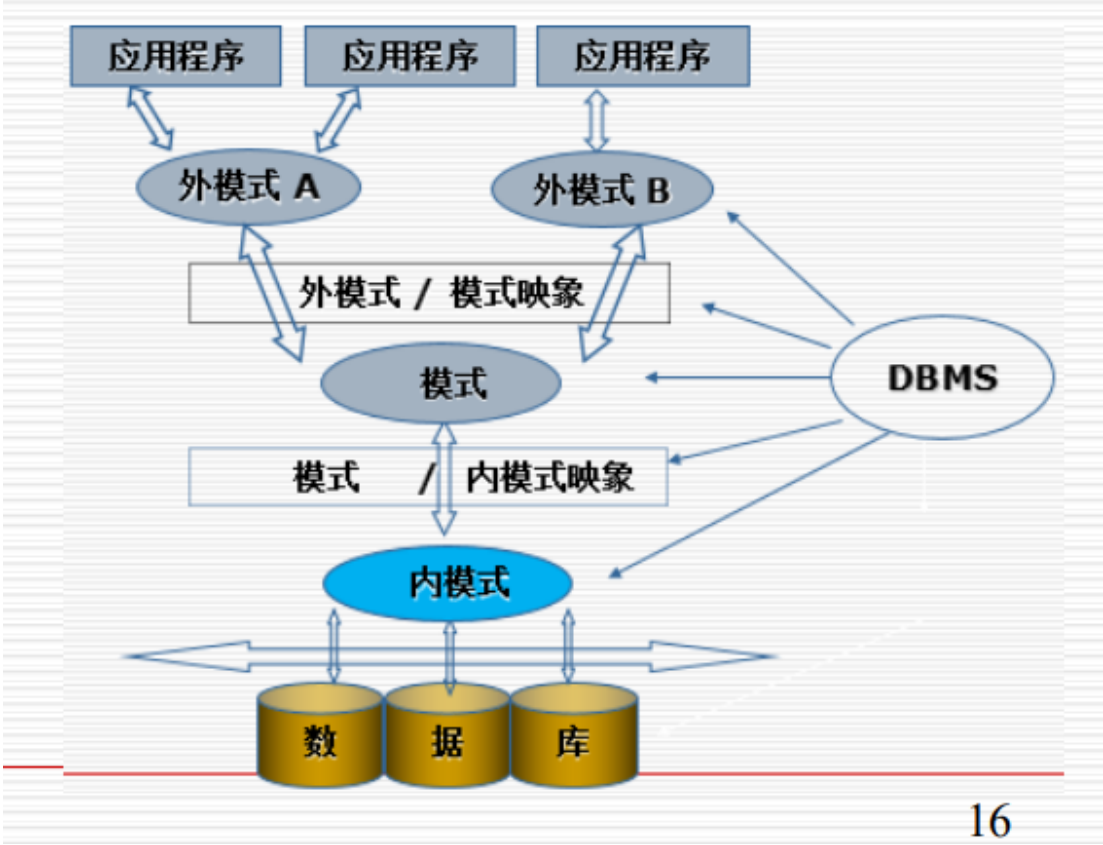
物理模型：

数据在外存上的存储组织方式，与存储介质的存储 性能、外存分配形式密切相关

模式

一个数据库只有一个模式

模式分为 外模式、模式、内模式



外模式和逻辑、特征相关，内模式和数据物理存储相关

外模式/模式映像：保证了数据的逻辑独立性

模式/内模式映像：保证了数据的物理独立性

ch2

数据模型三要素

1、数据结构：如何描述事物和组织数据

2、数据操作：如何操作数据库中的数据

3、完整性约束：如何约束数据库中的数据及数据之间 的联系

域

域是一组具有相同数据类型的值的集合，例如【学号】域、【学生名称】域。

※笛卡尔积

> 给定若干域D_1、D_2、D_3，这些域可以有相同的，其笛卡尔积为：
> $D_1 \times D_2 \times \ldots \times D_n = \{(d_1, d_2, \ldots, d_n) | d_i \in D_i,\ i = 1, 2, \ldots, n\}$

·所有域的所有取值的所有组合

·不能重复

元组

笛卡尔积中的每一个元素$(d_1, d_2, \ldots, d_n)$叫做一个n元组

分量

如上的每一个d_i称作一个**分量**

关系

> 笛卡尔积 $D_1 \times D_2 \times \ldots \times D_n$ 的子集叫做作用在域
> D_1、D_2、$\ldots$、D_n 上的关系，记作：
> $R(D_1、D_2、\ldots、D_n)$

R：关系名

n：关系的目或度

码

> 唯一标识一个元组的属性或属性组

·候选码、主码

·主属性、非主属性

任意两个元组的候选码不能相同

关系的完整性

1、实体完整性

2、参照完整性

3、用户定义完整性

> 实体完整性：若属性A为关系R的**主属性**，则属性A**不能为空**
> （主属性不能为空）

外码

设 F是基本关系R的一个或一组属性，但不是关系R的码

若 F是关系S的主码（或与其主码K对应）

则 F是关系R的**外码（FK）**

R为**参照关系**，S为**被参照关系**

> 参照完整性：
> （外码必须与参照关系对应）
> 若 F是基本关系R的外码，且与基本关系S的主码K相对应，

> 则 R中每个元组的F值
>
> - 要么为空值（全空）
> - 要么等于S中某个元组的主码值

ch3

关系代数

笛卡尔积：$R \times S = \{t_r t_s | t_r \in R \wedge t_S \in S\}$

选择：$\sigma_F(R) = \{t | t \in R \wedge F(t) = true\}$

投影：$\pi_A(R) = \{t[A] | t \in R\}$

选择和投影的区别：

选择是选择表中某几行，即具有全部属性的、符合条件的某几个实体。

投影是选择表中某几列，即选择所有实体的某几种属性，单拿出来作新的投影表

等值连接：
$R \underset{A=B}{\bowtie} S = \{t_r t_s | (t_r[A] = t_s[B]) \wedge (t_r \in R \wedge t_s \in S)\}$

这里的A和B代表 表R 和 表S 的某个对应属性

自然连接：$R \bowtie S$

指的是表R和表S所有相同且对应的属性均合并

外连接：

$R \overset{\bowtie}{} S$ 左外连接：表示S外连接到R，R不变，将S对应相等值并入R

$R \overset{}{\bowtie} S$ 右外连接：表示R外连接到S

除：$R \div S = T$

T中包含所有在R中但不在S中的属性及值，且**T的元组与S的元组所有组合都在R中**

ch6

E-R图

实体型：矩形

属性：椭圆形

联系：菱形

概念结构设计两条准则：

1、属性不能再具有需要描述的性质。

2、属性不能与其他实体具有联系。

数据依赖

数据依赖：关系内部属性与属性之间的一种约束关系

> 包括 函数依赖、多值依赖、连接依赖、包含依赖、其他

函数依赖：可以理解为$y = f(x)$，即表中的某列属性X唯一决定列属性Y，记作$X \rightarrow Y$.

非平凡函数依赖：$X \rightarrow Y$但$Y \not\subseteq X$（X不包含Y），称其为非平凡的函数依赖

完全函数依赖：若$X \rightarrow Y$，但对于X的任何真子集X`都不成立$X \not\rightarrow Y$，则称Y **完全函数依赖** 于X，否则称作 **部分函数依赖** 于X

传递函数依赖：若$X \to Y$、$Y \to Z$，且满足$Y \notin X$、$Y \nrightarrow Z$，则称Z对X传递函数依赖

范式

> 范式：R(U,F)符合x范式要求，则称R为x范式，记作$R \in xNF$。
> $1NF \supset 2NF \supset 3NF \supset BCNF \supset 4NF \supset 5NF$
> 后面的范式是在前边的范式基础上，消除了一些部分依赖关系。

第一范式 1NF

如果R(U,F)中所有属性都是**不可分的基本数据项。**则$R \in 1NF$

第二范式 2NF

$R \in 1NF$，且每一个**非主属性完全依赖于码。**则$R \in 2NF$

消解2NF方法：

1、【建立临时主码子集】把主码的每个子集都单拉出来作一个关系模式。

2、【非主属性各找各家】将完全依赖于主码的属性放到对应关系模式中。

3、【删除冗余主码子集】把只有主码的关系模式去掉。

第三范式 3NF

$R \in 2NF$，即在每一个非主属性**不部分依赖于码**的情况下，也**不传递依赖于码。**

消解3NF方法：

1、【提取中间依赖因子】：删除依赖于中间决定因子的所有属性。

2、【中间因子新建表】：新建一个关系模式，其中包含中间因子和依赖于该因子的所有属性

3、【中间因子设为主码】：将中间决定因子作为新关系的主码。

BCNF

$R \in 3NF$，且消除了主属性对其他属性的完全依赖。

SQL语句

数据库操作

```
/***------------------------------ 数据库操作 -----------
-------------------***/

-- 创建数据库 --
CREATE DATABASE $database_name;
-- 查询所有数据库 --
SHOW DATABASES;
-- 查询当前数据库 --
SELECT DATABASE();
-- 删除数据库 --
DROP DATABASE [IF EXISTS] $database_name;
-- 使用数据库 --
USE $database_name;
```

```
/***------------------------------ 表操作 -------------
------------------***/

-- 创建表 --
CREATE TABLE $table_name (
    $val_name_1 $type_1 NOT NULL ,/* 非空 */
    $val_name_2 $type_2 NOT NULL PRIMARY KEY ,/* 非空 主
键 */
    $val_name_3 $type_3 UNIQUE ,/* 此属性对每个实体唯一 */
    $val_name_3 $type_3 COMMENT $comment_3  ,/* 带注释
*/
    ...
    $val_name_n $type_n COMMENT $comment_n
) COMMENT $table_comment FOREIGN KEY($val_name_i)
REFERENCES $table_2($val_name_j);
/* val_name_i 为对应 table_2 上 $val_name_j 的外键 */

-- 修改表 --
ALTER TABLE $table_name $operate; /* 修改某个表 */
/* $operate 可选内容 */
ADD CONSTRAINT $consraint_name $consraint; /* 添加约束
*/
ADD CONSTRAINT $consraint_name PRIMARY KEY ($key_name);
/* 设置主键 */
ADD $val_name $type COMMENT $comment $constraint;/* 增
加表字段 */
MODIFY $val_name $new_type ; /* 修改变量类型 */
CHANGE $old_val_name $new_val_name $new_type COMMENT
$comment $constraint;/* 更改变量名及变量类型 */
DROP $val_name; /* 删除字段 */
RENAME TO $new_table_name; /* 修改表名 */

-- 删除表 --
DROP TABLE [IF EXISTS] $table_name; /* 删除表 */
TRUNCATE TABLE $table_name; /* 格式化某张表（删除内容，不删
除格式） */
```

```
/***------------------------------ 数据操作 ------------
------------------***/

-- 添加数据 --
INSERT INTO $table_name
VALUES($val_1,$val_2,...,$val_n); /*增加全部字段数据*/
INSERT INTO $table_name($val_name_1,$valname_4,...)
VALUES($val_1,$val_4,...); /* 增加指定字段数据 */
INSERT INTO $table_name($val_name_1,$valname_4,...)
VALUES($val_1,$val_4,...), ($val_1,$val_4,...),
($val_1,$val_4,...)/* 批量添加 */

-- 修改数据 --
UPDATE $table_name SET
$val_name_1=val_1,$val_name_1=val_1,...[WHERE
$condition];

-- 删除数据 --
DELETE FROM $table_name [WHERE $condition];


/***------------------------------ 查询语句 ------------
------------------***/
SELECT $字段  /* 要显示的内容 */
SELECT DISTINCT $字段 /* 合并重复内容 */
FROM $表      /* 要从那张表查询 */
WHERE $条件筛选 /* 要显示的内容具有什么属性（筛选） */
GROUP BY $分组字段
HAVING $分组字段条件筛选
ORDER BY $字段排序
INNER JOIN /* 内联合并 */
LEFT JOIN   /* 左连接 */
RIGHT JOIN  /* 右连接 */

-- 示例1: 字段查询 --
```

```
SELECT $val_name_1 AS $show_name_1,$val_name_2 FROM
$table_name

-- 示例2: 字段查询+去重 --
SELECT DISTINCT $val_name_1,$val_name_2 FROM
$table_name

-- 示例3: 条件查询 --
SELECT $val_name FROM $table_name WHERE $condition;
/* $condition可选内容 */
> >= < <= = != /* 等式|不等式 符号 */
BETWEEN $val_1 AND $val_2 /* 范围筛选 */
IN(...) /* 括号中可以为某个列表，或者某种筛选结果SELECT (临时
列表)  */
LIKE $占位符 /* 模糊搜索使用，‘_’ 匹配单个模糊字符，‘%’匹配多
个模糊字符 */
IS NULL/* 空 */
NOT NULL/* 非空 */

-- 示例4: 连接查询 --
SELECT $val_name,$table_1.$val_1 FROM $table_1 INNER
JOIN $table_2 ON $table_1.$val_1 = $table_2.val_1;
/* 根据 val_1 合并 table_1 和 table_2 ,并对val_name进行查
询，  */

-- 示例5: 嵌套查询 --
SELECT $val_name FROM $table_1 WHERE NOT EXISTS (SELECT
* FROM $table_2 WHERE $val_1 = $table_1.$val_2 AND
$val_2 = 'val');
/* 将table_1和table_2的val_1对应，并查询table_1中 不存在于
后者查询的内容 */

-- 聚散函数 --
SELECT $func($val_name_1,$val_name_2,...) FROM
$table_name;
/* $func可选内容 */
COUNT /* 总数统计 */
```

```
MAX /* 最大值 */
MIN /* 最小值 */
AVG /* 平均值 */
SUM /* 求和 */

-- GROUP BY --
SELECT $val_name,COUNT(*) AS $show_name FROM
$table_name WHERE $condition
GROUP BY $val_name HAVING COUNT(*)>4;
/*
    1、在$table_name对应表中，先对满足$condition条件的数据进
行筛选，
    2、对于通过筛选的数据，将其按照$val_name值 分为若干 子表
(group)
    3、统计每个子表的数据总数，筛选出总数大于4的子表
    4、输出通过筛选子表的对应$val_name和总数

    ※‘子表’ 是[GROUP BY]的抽象化理解
*/

-- 排序 --
SELECT $val_name FROM $table_name ORDER BY $val_name_i
$sort_func;
/* $sort_func可选内容 */
ASC /* 升序 */
DESC /* 降序 */


/***------------------------------ 权限操作 ------------
------------------***/
-- 增加权限 --
GRANT $permissions ON TABLE $table TO $user;
/* $permissions可选内容 */
CREATE SCHEMA
CREATE TABLE , ALTER TABLE
CREATE VIEW
CREATE INDEX
```

```
SELECT,INSERT,UPDATE,REFERENCES,ALL PRIVILEGES

-- 撤销权限 --
REVOKE $permissions ON TABLE $table TO $user;

/***------------------------------ 架构操作 ------------
------------------***/
/*
    一个数据库不允许存在同名的两个表，除非他们在不同的架构SCHEMA
里
    SCHEMA可以隔离命名空间，方便管理
*/
CREATE SCHEMA $schema AUTHORIZATION $user;
DROP SCHEMA $schema
```