开发成功数据库应用的要点

- 需要理解数据库体系结构
- 需要理解锁和并发控制特性
  - 每个数据库都以不同的方式实现
- 不要把数据库当“黑盒”
- 性能、安全性都是适当的被设计出来的
- 用尽可能简单的方法解决问题
  - “创造”永远追不上开发的步伐
- DBA和RD之间的关系

数据库体系结构的差异

- **不能把数据库当成“黑盒”使用，因为每个数据库都是非常不同的**
- Oracle和MySQL的差别，类似
  - Windows和Linux的差别
  - iOS和Android的差别
  - 虽然都是DBMS，但它们也有相当的差异
- 了解这种差异，了解你所使用数据库的特性，是开发成功数据库应用的基础

并发控制的问题

- 现实存在并发，我们需要保持数据的一致性，所以要做并发控制
- 锁机制，使得并发控制成为可能
- **不同的数据库，实现锁机制是不一样的**

比如：Oracle的锁机制就比较特别

- 这个时候，Transaction1能提交吗?

时序	Transaction 1	Transaction 2
T1	Begin	
T2	Read（X）x=1000	Begin
T3		Read（X）x=1000
T4		Write（X）x=x-800=200
T5		Submit
T6	**Submit**	

Oracle存在有时读不到正确数据的现象

- Oracle的多版本控制，读一致性的并发模型
  - 读一致查询：对于一个时间点（point time），查询会产生一致的结果
  - 非阻塞查询：查询不会被写入器阻塞，但在其它数据库中可能不是这样的

有一个最简单的方式演示Oracle中的多版本：

```
Create table t
As
Select * from all_users
/

Variable x refcursor
Begin
Open :x for select * from t;
End;
/
```

```
Declare

Begin

Delete from t;

Commit;

End;

/
```

Oracle这种锁机制的好处是什么？

- *例子：ACCOUNTS（account_number, account_balance）一个银行的账户余额*

  为了简单，只考虑一个仅有四行的表（同时假设每个数据库块中只存放一行数据。）

  帐号，余额

  1；500

  2；250

  3；400

  4；100

- 我们想运行一个日报表，了解银行里有多少钱。下面是一个非常简单的查询。
  - Select sum(account_balance) from accounts;

Oracle和其它数据库在并发上的差别

帐号，余额
1；500
2；250
3；400
4；100

Oracle的做法

	R&W	W
T1	Read（1）	
T2	Read（2）	Write（1）500-400
T3	Read（3）	Write（4）100+400
T4	**Read（4）**	commit
T5	Sum（1+2+3+4）	
T6	**Commit**	

其它数据库的做法

	R&W	W
T0	**Lock（T）**	
T1	Read（1）	
T2	Read（2）	Write（1）500-400
T3	Read（3）	Write（4）100+400
T4	**Read（4）**	Commit **（Waiting and Roll Back）**
T5	Sum（1+2+3+4）	
T6	**Commit**	Roll Back

从体系结构和特性中了解具体数据库的锁机制

- 比如Oracle实现的锁机制
  - 只有修改才加行级锁
  - Read绝对不会对数据加锁
  - Writer不会阻塞Reader
  - 读写器绝对不会阻塞写入器

课后思考

- 对大多数码农而言，数据库锁机制好像都是自动和透明实现的，那么深入了解每个数据库的锁机制实现细节，对码农编码有什么影响嘛？
- 根据这节课Oracle的锁机制特征的分析，你尝试去了解一下MySQL、SQLServer这些其它数据库锁机制实现的特征

- **请将你的思考和分析，完成课后的小作业（不是两次大作业），优秀的作业会予以表扬或其它奖励**

End

休息一下，等待下一节视频