## ORACLE浅析

1. 事务

事务是一个完整工作的逻辑单元，一个事务由一个或多个SQL组成。通过事务机制确保一组SQL的操作，要么都成功执行，要么一个也不执行。从而保证数据的一致性。

特点：

1. 原子性：一个事务中的所有SQL是一个整体，要么都执行成功，要么都不成功。
2. 一致性：数据的一致性，事务结束后，数据应该和期望的数据一致。
3. 隔离性：防止事务并发执行时，造成的数据不一致状态。
4. 持久性：事务结束后，对数据库中的数据的影响是永久的。

数据异常：

1. 脏读
2. 不可重复读
3. 幻读
4. 隔离级别

事务中遇到的异常与事务的隔离级别设置有关，隔离级别越高，异常越少，并发效果越底。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 隔离级别 | 描述 | 脏读 | 不可重复读 | 幻读 |
| Read uncommited | 读未提交 | √ | √ | √ |
| Read commited | 读已提交 | × | √ | √ |
| Repeated read | 可重复读 | × | × | √ |
| Serializable | 串行读 | × | × | × |

Oracle支持：read commited、serializable，并增加了read only、read write。

1. 对事务设置

SET TRANSACTION ISOLATION\_LEVEL READ COMMITED|SERIALIZABLE;

SET TRANSACTION READ ONLY|READ WRITE;

1. 对会话设置

ALTER SESSION SET ISOLATION\_LEVEL = READ COMMITED|SERIALIZABLE;

1. 锁模式

Lock table in share mode/row share mode/exclusive mode/row exclusive mode/share row exclusive mode;

Serializable：当设置事务为串行时，记录当前的SCN，所有其他事务的操作都不可见，且如果当前事务尝试修改其他事务已经修改的内容会报错。如果其他事务修改了，但没提交，当前事务会挂起，当其他事务提交时，当前事务则立即报错。

1. 索引

如字典的目录，可以根据目录中的页码快速找到需要的内容。

1. 应该创建索引的字段

① 经常作为查询条件的字段

② 经常用作多表连接的字段

③ 经常需要排序的字段

1. 不应该货应少创建索引的字段

① 表数据少

② 经常DML的表

③ 数据重复且分布均匀

④ 查询中很少用到的字段

1. 索引失效的情况

① 不要在索引列中用is null或is not null，改为 column >=|<= 一个绝对没有值，如chr(0)；

② 不要使用函数，如substr(name,1,3) = ‘ABC’，改为name like ‘ABC%’，避免通配符出现在词首；

③ 不要使用NOT、!=、<>，改为> or <；

④ 不要计算 c/10 >10、c>10/10；

1. 索引类型

① b树索引：最常见的索引，根据ROWID快速定位访问行

② 位图索引：适用于基数比较少的列

③ 降序索引：默认的b树是升序 column desc变成降序

④ 函数索引：索引是某一列或多列的某个表达式，只有查询语句是一样的表达式时，索引才生效

⑤ 反转索引：反转了b树的字节码，使索引条目更均匀，但不能基于快速查询了

⑥ 分区索引：本地索引、全局索引（索引列第一列必须是前缀列）

1. 扫描类型

① 全表扫描

② ROWID扫描：访问的列都在索引中

③ 索引范围扫描

1. SQL优化

1）执行顺序：FROM、WHERE、GROUP BY、HAVING、ORDER BY；

2）FROM子句是从右往左执行的，将过滤数最大的放在后面，表连接放在最前面；

3）避免使用\*，oracle会将\*依次转换为列名，通过字典，花时间；

4）对多列索引，用OR会造成全表扫描，用UNION代替

* + 1. Select \* from table1 where id = 10 or name = ‘a’;--低效
    2. Select \* from table1 where id = 10 union select \* from table1 where name = ‘a’;--高效
    3. Table1表有id和name两个索引

5）用EXISTS替代IN，NOT EXISTS替代 NOT IN

① Select \* from t1 where exists (select 1 from t2 where t2.a = t1.a);

② Select \* from t1 where t1.a in(select t2.a from t2);

③ Exists先运行主查询，得到t1的结果集，再用结果集去子查询中匹配，匹配成功则返回；

* + 1. In先运行子查询，得到t2结果集，在去主查询匹配；
    2. 当主表<<子表时用exists好，反之用in好；

6）用EXISTS代替DISTINCT

① Select \* from distinct deptn,name from dept d,emp e where d.deptn=e.deptn;

② Select \* from deptn,deptName from dept d where exists(select 1 from emp e where e.deptn = d.deptn);

③ 1、2不能用左连接，因为要求只有两表都有，且相等的时候才显示，而左连接显示基表所有。

④ 1是将两表关联，两表都进行了全表扫描；2是先将部门表结果集扫描出来，在拿它去员工表作查询，如果查到立即返回，只查了后面一个表的一部分。

7）删除重复记录用ROWID方式

Delete from emp e where e.rowid >(select min(x.rowid) from emp x where x.empn=e.empn);

8）当删除全表，且不考虑恢复时，用truncate替代delete。

9）Having子句，是在分组之后，检索出所有记录后，对结果集进行过滤，需要排序、总结。

10）Group by与having一起用很强大

Select name,min(score) score from student group by name having score >80;

11）用>=替代>，num>=4，可以直接定位到第一个为4的，而num>3，先定位到3，再扫描找第一个3。

12）DECODE函数替代if else

DECODE(字段，比较1，值1，比较2，值2，...);

1. SQL工作流程
2. 用户发出SQL请求，打开游标；
3. 把SQL语句、语法分析，执行计划，数据字典等信息存入内存中的共享池内；
4. 从数据文件中把相关数据块读入数据缓冲区；
5. 做相应操作，若做修改，先加上行级锁，经确认后，把改过前后记录内容存入重做日志和回退段中；
6. 返回结果给用户，关闭游标。

注：SQL语句大小写敏感，同样的一个语句，若大小写不同，分析两次，建议大小。

1. oracle乐观锁与悲观锁
2. 悲观锁：每次取数据（select）的时候都会认为别人会修改，所以每次拿数据的时候都会上锁。
3. select for update;--别人修改时，等待
4. Select for update nowait;--别人修改时，报错
5. 乐观锁：oracle默认使用乐观锁，select \* from table;
6. 自定义使用悲观锁

begin

Lock table in share mode；

操作！

End;

1. 关系型与非关系型数据库
2. 关系型：数据是以二维表格形式存储和呈现的
3. 关系：二维表，即关系名，表名
4. 元组：二维表中的一行
5. 属性：二维表中的一列，字段
6. 域：属性的取值范围
7. 关键字：主键
8. 关系型优点
9. 容易理解：二维表符合逻辑
10. 使用方便：通用的SQL语言使操作方便
11. 易于维护：丰富的完整体，大大降低了数据冗余和不一致的概率
12. 关系型瓶颈
13. 高并发读写需求：网站的用户并发量高，硬盘I/O是个瓶颈
14. 海量数据读写：在包含海量数据的表中查询，效率低
15. 高扩展性与可用性：难横向扩展，停机维护需要数据迁移
16. 关系型特性不需要了
17. 事务一致性：不需要很高的一致性，对网站用户来说
18. 读实时性：
19. 复杂SQL
20. 非关系型
21. 面向高性能并发读写的KEY-VALUE数据库
22. 面向海量数据访问的面向文档数据库，快速查询
23. 面向可扩展的分布式数据库
24. 关系型数据库具有固定的表结构，因此扩展性差，而现在的需求变化很快，数据库结构变化巨大，关系型数据库难以应付，所以非关系型来了
25. Oracle与MySQL
26. 并发性
27. MySQL以表级锁为主，innoDB有行级锁，但必须基于索引
28. Oracle使用行级锁，并发性好很多
29. 一致性
30. oracle支持read commited和serializable隔离级别，可以用最高的读一致性。用UNDO表空间的回退段和重做日志来控制版本。
31. MySQL没有多版本控制，只支持read commited，一个session读取数据时，其他session不能修改，但可以在表最后插入数据。更新时，其他session无法访问数据。
32. 事务
33. oracle完全支持事务
34. MySQL在innoDB引擎的行级锁下才支持
35. 数据的持久性
36. oracle可恢复
37. Mysql可丢失
38. 提交方式
39. oracle手动提交
40. MySQL自动提交
41. 逻辑备份
42. oracle备份时，不锁定数据且数据是一致的，通过版本控制
43. MySQL备份时，需要锁定数据，影响dml使用
44. 热备份
45. oracle热备份是，不影响使用数据库，可在归档日志和重做日志进行一致的恢复
46. MySQL需要加锁