# **简介**

1.说明

　　1）索引是数据库对象之一，用于加快数据的检索，类似于书籍的索引。在数据库中索引可以减少数据库程序查询结果时需要读取的数据量，类似于在书籍中我们利用索引可以不用翻阅整本书即可找到想要的信息。

　　2）索引是建立在表上的可选对象；索引的关键在于通过一组排序后的索引键来取代默认的全表扫描检索方式，从而提高检索效率

　　3）索引在逻辑上和物理上都与相关的表和数据无关，当创建或者删除一个索引时，不会影响基本的表；

　　4）索引一旦建立，在表上进行DML操作时（例如在执行插入、修改或者删除相关操作时），oracle会自动管理索引，索引删除，不会对表产生影响

　　5）索引对用户是透明的，无论表上是否有索引，sql语句的用法不变

6）oracle创建主键时会自动在该列上创建索引

# ****索引分类****

**1. B树索引**（默认索引，保存经过排序后的索引列和对应的rowid值）

1）说明：

　　1.oracle中最常用的索引；B树索引就是一颗二叉树；叶子节点（双向链表）包含索引列和指向表中每个匹配行的ROWID值

　　2.所有叶子节点具有相同的深度，因而不管查询条件怎样，查询速度基本相同

　　3.能够适应精确查询、模糊查询和比较查询

2）分类：

　　 UNIQUE,NON-UNIQUE(默认),REVERSE KEY（数据列中的数据是反向存储的）

3）创建例子

craete index index\_sno on student('sno');

4）适合使用场景：

　　列基数（列不重复值的个数）大时适合使用B数索引

**2. 位图索引**

1）说明：

　1.创建位图索引时，oracle会扫描整张表，并为索引列的每个取值建立一个位图（位图中，对表中每一行使用一位（bit，0或者1）来标识该行是否包含该位图的索引列的取值，如果为1，表示对应的rowid所在的记录包含该位图索引列值），最后通过位图索引中的映射函数完成位到行的ROWID的转换适合与决策支持系统；   
位图索引做UPDATE代价非常高； 非常适合OR操作符的查询； 基数比较少的时候才能建位图索引；

2)创建例子

create bitmap index index\_sno on student(sno);

3) 适合场景：

对于基数小的列适合简历位图索引（例如性别等）

**3.单列索引和复合索引**（基于多个列创建）

1) 注意：

　　即如果索引建立在多个列上，只有它的第一个列被where子句引用时，优化器才会使用该索引，即至少要包含组合索引的第一列

**4. 函数索引**

1)说明：

　　1. 当经常要访问一些函数或者表达式时，可以将其存储在索引中，这样下次访问时，该值已经计算出来了，可以加快查询速度

　　2. 函数索引既可以使用B数索引，也可以使用位图索引；当函数结果不确定时采用B树索引，结果是固定的某几个值时使用位图索引

　　3. 函数索引中可以用len、trim、substr、upper（每行返回独立结果），不能使用如sum、max、min、avg等

2）例子：

create index fbi on student (upper(name));select \* from student where upper(name) ='WISH';

**5. 分区索引和全局索引**

这2个是用于分区表的时候。前者是分区内索引，后者是全表索引

5. 反向索引（REVERSE）

这个索引不常见，但是特定情况特别有效，比如一个varchar(5)位字段(员工编号)含值（10001,10002,10033,10005,10016..）

这种情况默认索引分布过于密集，不能利用好服务器的并行

但是反向之后10001,20001,33001,50001,61001就有了一个很好的分布，能高效的利用好并行运算。

**6.HASH索引**

HASH索引可能是访问数据库中数据的最快方法，但它也有自身的缺点。集群键上不同值的数目必须在创建HASH集群之前就要知道。需要在创建HASH集群的时候指定这个值。使用HASH索引必须要使用HASH集群。

# ****索引建立原则总结****

　　1. 如果有两个或者以上的索引，其中有一个唯一性索引，而其他是非唯一，这种情况下oracle将使用唯一性索引而完全忽略非唯一性索引

　　2. 至少要包含组合索引的第一列（即如果索引建立在多个列上，只有它的第一个列被where子句引用时，优化器才会使用该索引）

　　3. 小表不要简历索引

　　4. 对于基数大的列适合建立B树索引，对于基数小的列适合简历位图索引

　　5. 列中有许多空值，不适合建立索引，但经常查询该列上非空记录时应该建立索引

　　6. 经常进行连接查询的列应该创建索引

　　7. 使用create index时要将最常查询的列放在最前面

　　8. LONG（可变长字符串数据，最长2G）和LONG RAW（可变长二进制数据，最长2G）列不能创建索引

9.限制表中索引的数量（创建索引耗费时间，并且随数据量的增大而增大；索引会占用物理空间；当对表中的数据进行增加、删除和修改的时候，索引也要动态的维护，降低了数据的维护速度）

10.经常检索排序大表中40%或非排序表7%的行，建议建索引；

# **索引使用（创建、修改、删除、查看）**

**1.创建索引语法**

CREATE [UNIQUE] | [BITMAP] INDEX index\_name --unique表示唯一索引

ON table\_name([column1 [ASC|DESC],column2 --bitmap，创建位图索引

[ASC|DESC],…] | [express])

[TABLESPACE tablespace\_name]

[PCTFREE n1] --指定索引在数据块中空闲空间

[STORAGE (INITIAL n2)]

[NOLOGGING] --表示创建和重建索引时允许对表做DML操作，默认情况下不应该使用

[NOLINE]

[NOSORT]; --表示创建索引时不进行排序，默认不适用，如果数据已经是按照该索引顺序排列的可以使用

**2.修改索引**

1）重命名索引

alter index index\_sno rename to bitmap\_index;

2) 合并索引（表使用一段时间后在索引中会产生碎片，此时索引效率会降低，可以选择重建索引或者合并索引,合并索引方式更好些，无需额外存储空间，代价较低）

alter index index\_sno coalesce;

3)重建索引

　　方式一：删除原来的索引，重新建立索引

　　方式二：alter index index\_sno rebuild;

**3.删除索引**

drop index index\_sno;

1. **查看索引**

select index\_name,index-type, tablespace\_name, uniqueness from all\_indexes where table\_name ='tablename';

-- eg:

create index index\_sno on student('name');

select \* from all\_indexes where table\_name='student';

# **注意事项**

1. 默认的情况下，当我们建立一个unique constraint的时候，Oracle会自动的建立一个同名的unique index，真正的唯一性验证是通过索引完成的。当删除unique constraint的时候，自动建立的unique index也会被删除。
2. 如果在字段上手工建立了一个unique index，名称无所谓。 在相同字段上建立unique constraint 的时候，Oracle会使用已经存在的index来进行验证，而不会再次建立index。 当删除unique constraint的时候，手工建立的unique index不会被删除。 这时插入数据的时候仍然会报唯一性冲突

------------自动建索引------------------------------

drop table test\_ak1 purge;

create table test\_ak1 (x int, y int, z int);

-- 直接建立unique constraint,默认建立一个同名unique index alter table test\_ak1 add (constraint ak\_1 unique (x));

-- 查看约束和索引

select c.constraint\_name, c.constraint\_type, c.index\_name from user\_constraints c where c.table\_NAME = upper('test\_ak1');

select i.index\_name,i.uniqueness from user\_indexes i where i.table\_name = upper('test\_ak1');

-- 删除约束的同时会删除索引

alter table test\_ak1 drop constraint ak\_1;

---------手动建索引---------------------------------

drop table test\_ak2 purge;

create table test\_ak2 (x int, y int, z int);

-- 手工建立unique index create unique index ak\_2 on test\_ak2 (x);

-- 建立一个unique constraint，会使用相同字段上的unique index，不论名字是否一样

alter table test\_ak2 add (constraint ak\_2 unique (x));

-- 查看约束和索引

select c.constraint\_name, c.constraint\_type, c.index\_name from user\_constraints c where c.table\_NAME = upper('test\_ak2');

select i.index\_name,i.uniqueness from user\_indexes i where i.table\_name = upper('test\_ak2');

-- 直接删除索引，会报ora-02429错误 drop index ak\_2;

-- 删除约束的时候，索引不受影响 alter table test\_ak2 drop constraint ak\_2;

1. 条件字段选择性弱，查出的结果集较大，不走索引；
2. 强制索引:/\*+INDEX(表名,索引名字)\*/ 的方法,但是这种方法有个缺点，如果索引的名字改变了，就会导致程序重新改动，大大增加维护成本。 索引提示还可以使用列的方法进行,语法是/\*+INDEX(表名,(索引列的列表))\*/,这种方法即使索引的名字变了，也不会导致程序重新改动
3. where条件等号两边字段类型不同，不走索引；
4. 优化器分析的统计信息陈旧也可能导致不走索引；

execute dbms\_stats.gather\_table\_stats($USER\_NAME,$TABLE\_NAME);

1. 索引字段 is null 不走索引；
2. 对于count(\*)当索引字段有not null约束时走索引，否则不走索引；
3. like 后面的字符当首位为通配符时不走索引；
4. 使用不等于操作符如：<>、!= 等不走索引；
5. 如果表中有两个以上（包括两个）索引，其中有一个唯一性索引，而其他是非唯一性，在这种情况下，ORACLE将使用唯一性索引而完全忽略非唯一性索引．
6. 查询有null值不走索引解决方案--创建该字段和一个常量的联合索引即可

create index idx\_test\_null\_1 on tableName(id,1);

查询空值

select count(\*) from tableName where id is null; --走索引idx\_test\_null\_1 range scan

1. 避免在索引列上使用函数
2. 模糊搜索时避免使用前置通配符
3. 避免在索引列上使用not
4. 避免在索引列上自动转换,数据类型要和数据库保持一致
5. 假设EMP\_TYPE是一个字符类型的索引列.

SELECT USER\_NO,USER\_NAME,ADDRESS FROM USER\_FILES WHERE USER\_NO = 109204421

这个语句被ORACLE转换为:SELECT USER\_NO,USER\_NAME,ADDRESS FROM USER\_FILES WHERE TO\_NUMBER(USER\_NO) = 109204421

因为内部发生的类型转换, 这个索引将不会被用到!

1. 小表尽量避免使用索引，在扫描小表时,使用索引同样虽然能提高效率,但是每当有记录在表中增减或索引列被修改时, 将为此多付出4 , 5 次的磁盘I/O 。此外索引需要额外的存储空间和处理时间,有些不必要的索引反而会使查询反应时间变慢。索引需要定期维护重建
2. 避免在索引列上使用计算
3. 通过使用>=、<=等，避免使用NOT命令，用>=,<=代替>,<,!=,<>,not......
4. 创建了索引并不一定就会使用，oracle自动统计表的信息后，oracle的优化器根据这些信息决定是否使用索引及使用哪个索引，oracle的优化器主要有RBO和CBO，RBO是基于规则的优化，而CBO是基于成本的优化，现在oracle主要是CBO为主

查看表分析信息：

1) select \* from all\_tables where table\_name='TEMP\_TABLE';（表名一定要大写）

如果对该表没有做过统计分析，则查询的大部分栏位值都为空

2) analyze table tableName compute statistics;执行时间与表的数据量有关

3) select \* from all\_tables where table\_name=' TEMP\_TABLE ';之前为空的部分字段，现在查询已经有值

1. 当查询的返回的数据占总量数据的百分比小于20%时，建索引才有效果
2. 不是主键的索引值可以为空，主键索引不能为空。
3. 经常与其他表进行连接的表，在连接字段上应该建立索引。
4. 使用索引的时候需要注意：

　　a）能用唯一索引，一定用唯一索引

　　b）能加非空，就加非空约束

　　c）一定要统计表的信息，索引的信息，柱状图的信息。

　　d）联合索引的顺序不同，影响索引的选择，尽量将值少的放在前面

1. 在索引列上使用函数。如SUBSTR,DECODE,INSTR等，对索引列进行运算.需要建立函数索引就可以解决了。
2. 使用<>、not in 、not exist，对于这三种情况大多数情况下认为结果集很大，一般大于5%-15%就不走索引而走FTS。
3. 当变量采用的是times变量，而表的字段采用的是date变量时.或相反情况不走索引
4. 索引失效，可以考虑重建索引，rebuild online。
5. B-tree索引 is null不会走,is not null会走（Oracle的B-Tree索引不包含null，is not null正好限制了数据是在索引中的）,位图索引 is null,is not null 都会走、联合索引 is not null 只要在建立的索引列（不分先后）都会走。



Sql优化案例

https://blog.csdn.net/yws\_1990/article/details/51296515