介绍几个其它类型的索引

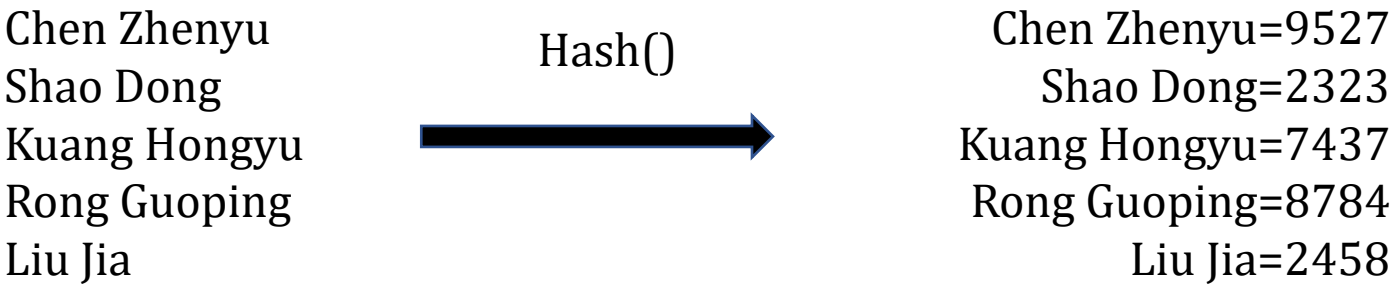
- 哈希索引（Hash Index）： MySQL

- 位图索引（Bitmap Index）： Oracle
- 位图联结索引（Bitmap join index）:Oracle

- 函数索引（function-based index）

哈希索引

- 哈希索引结构：



- 根据结构，你能告诉我 哈希索引能做什么不能做什么？
  - 碰撞率的问题

位图索引

- Bitmap index，Oracle7.3引入，位数据库仓库查询环境设计
- 位图索引的结构

Table 11-6. *Representation of How Oracle Would Store the* JOB-IDX *Bitmap Index*

Value/Row	1	2	3	4	5	6	7	8	9	10	11	12	13	14
ANALYST	0	0	0	0	0	0	0	1	0	1	0	0	1	0
CLERK	1	0	0	0	0	0	0	0	0	0	1	1	0	1
MANAGER	0	0	0	1	0	1	1	0	0	0	0	0	0	0
PRESIDENT	0	0	0	0	0	0	0	0	1	0	0	0	0	0
SALESMAN	0	1	1	0	1	0	0	0	0	0	0	0	0	0

Select count(*) from emp where job =’CLERK’ or job = ‘MANAGER’

Select * from emp where job =’CLERK’ or job = ‘MANAGER’

什么时候该使用位图索引

- 相异基数（distinct cardinality）低
- 大量临时查询的聚合

假设你有一个很大的表，T(gender, location, agegroup)
Gender M F
Location 1-50
Agegroup 18 and under 19-25 26-30 31-40 41 and over
而你又必须支持大量临时查询

```
Select count(*)
From t
Where gender = ‘M’  And location in (1, 10,30)
  And agegroup = ’41 and over’
```

```
Select *
From t
Where (  (gender = ‘M’ and location =20)
     Or (gender = ‘F’ and location =22)
  And agegroup = ’18 and under’;
```

```
Select count(*) from t where location in (11,20,30);
Select count(*) from t where agegroup = ’41 and over’ and gender = ‘F’;
```

位图联结索引（bitmap join index）

- 允许使用另外某个表的列对一个给定表建立索引。实际上，这就是允许对一个索引结构（而不是表本身）中的数据进行逆规范化。

Emp(empid,deptno) Dept(Deptno, Dname)

```
Select count(*)
From emp,dept
Where emp.deptno = dept.deptno
And dept.dname = ‘SALES’

Select emp.*
From emp,dept
Where emp.deptno = dept.deptno
And dept.dname = ‘SALES’
```

```
Create bitmap index emp_bm_idx on emp(d.dname)
From emp e, dept d
Where e.deptno = d.deptno
```

MySQL怎么办?

- MySQL没有位图索引，1）优化替代索引组合；2）低选择性添加特殊索引
- Select * from profiles where sex = ‘M’ order by rating limit 10;
  - 可以添加sex， rating列上的复合索引。
- select * from profiles where sex = ‘M’ order by rating limit 100000, 10;
  - 依旧很慢，更好的策略是限制用户查看的页数
  - 也可以：

```
Select * from t inner join (
    Select id from t
    Where x.sex = ‘m’ order by rating limit 100000, 10
)AS x USING id;
```

函数索引

- 函数索引，对F(x)的值构建索引，在通过对索引读取x所指向的记录行
  - X索引，和F(x)的索引完全不一样
- 想一想，函数索引能用在哪？
  - 不区分大小写的查询
  - T、F的巨大差异下的索引
  - 有选择的唯一性

```
Creat index emp_upper_idx on emp(upper (ename))
Select * from emp where upper(name) = 'KING'
```

```
Create unique index active_project_must_be_unique on projects(case when status = 'ACTIVE' then name end)
```

还有很多的其它索引，需要自己学习

- 首先，现看索引的**结构**，从结构-能做什么-不能做什么-练习，再循环

- 思考题
  - 请尝试，构建一个本课相似的例子（比如本课程的例子、电脑的配置的例子等等）插入大量数据，在MySQL上，尝试用B树索引模拟位图索引的功能。
  - 请再想想，还有什么场景下可以使用函数索引或者哈希索引?
  - **欢迎你在作业区留言，期待你的留言~**

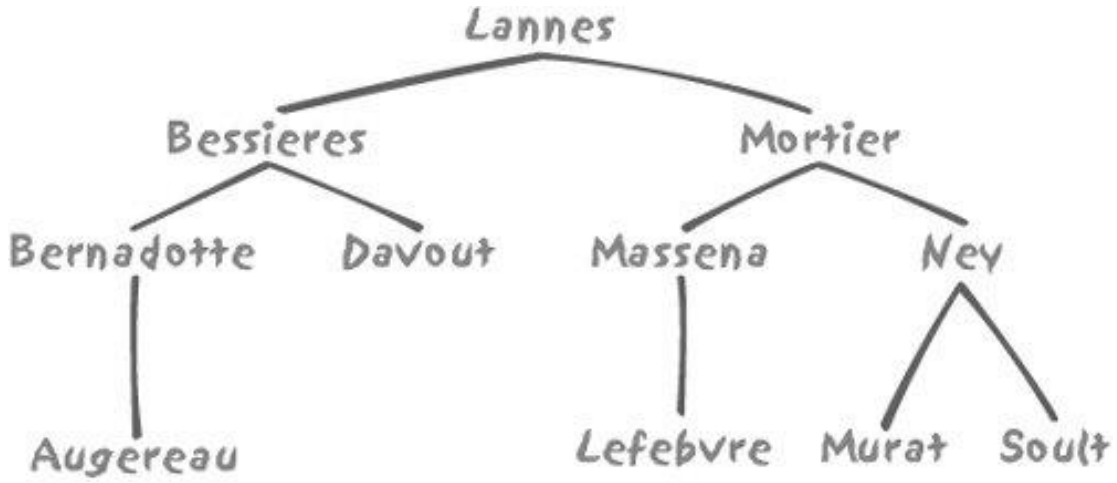
索引使用的典型问题

- 函数和类型转化对索引的影响
- 索引和外键
- 同一个字段，多个索引
- 系统生成键
- 总结，为什么没有使用我的索引?

函数和类型转换对索引的影响

- Where f (indexed_col) = ‘ some value‘
- 这种检索条件会使索引无法发挥作用
  - 日期函数
  - 隐式类型转换

字符串和日期的例子



where name = 'MASSENA'

where substr(name, 3, 1) = 'R'

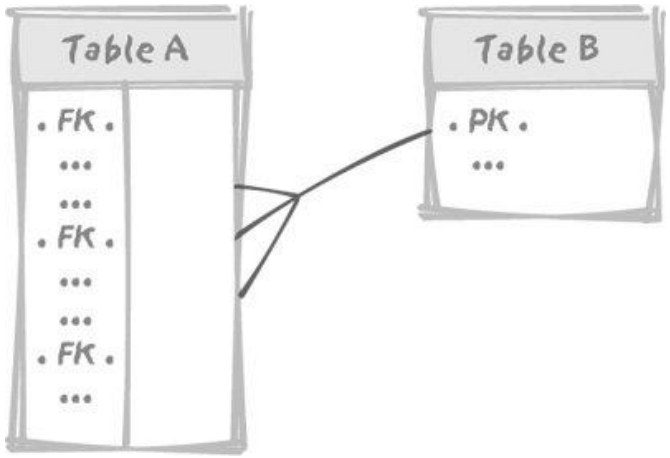
where date_entered = to_date('18-JUN-1815', 'DD-MON-YYYY')

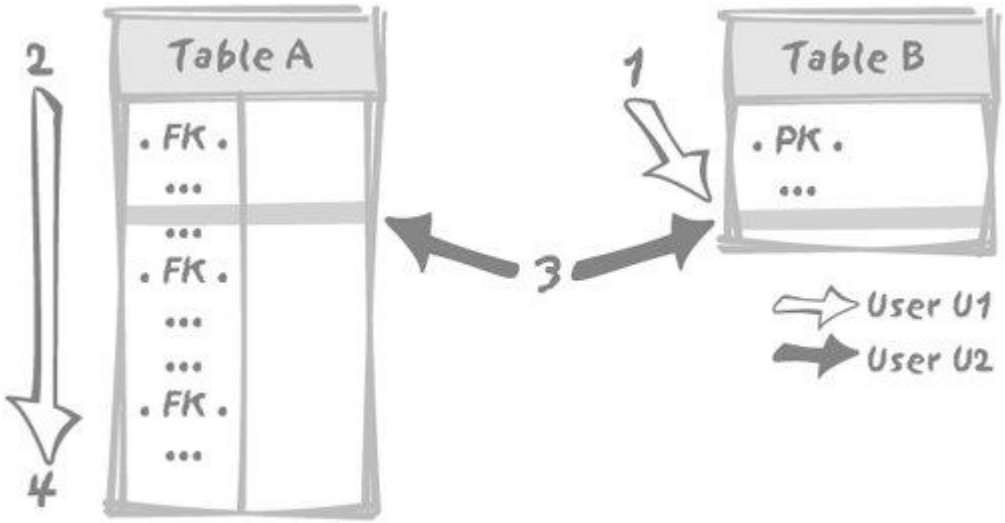
where trunc(date_entered) = to_date('18-JUN-1815', 'DD-MON-YYYY')

where date_entered >= to_date('18-JUN-1815', 'DD-MON-YYYY') and
date_entered < to_date('19-JUN-1815', 'DD-MON-YYYY')

索引与外键

- 系统地对表的外键加上索引的做法非常普遍
  - *但是为什么呢?*
  - *有例外吗?*
- 建立索引必须有理由
  - 无论是对外键，或是其他字段都是如此





同一字段，多个索引

- 如果系统为外键自动增加索引，常常会导致同一字段属于多个索引的情况

Order_Details (Order_id,article_id)

Order_Details (article_id,Order_id,)

当不需要为Article_id构建索引的时候，会引入额外索引

- 为每个外键建立索引，可能会导致多余索引

系统生成键

- 系统生产序列号，远好于
  - 寻找当前最大值并加1
  - 用一个专用表保存”下一个值“且加锁更新
- 但如果插入并发性过高，在主键索引的创建操作上会发生十分严重的资源竞争
- 解决方案
  - 反向键索引或叫逆向索引（reverse index）
  - 哈希索引（hash indexing）

为什么没有使用我的索引?

- 情况1：我们在使用B+树索引，而且谓词中没有使用索引的最前列
  - T，T(X,Y)上有索引，做SELECT * FROM T WHERE Y=5
- 跳跃式索引（仅CBO）

为什么没有使用我的索引？（cont’）

- 情况2：使用SELECT COUNT(*) FROM T，而且T上有索引，但是优化器仍然全表扫描
- 情况3：对于一个有索引的列作出函数查询
  - Select * from t where f(indexed_col) = value
- 情况4：隐形函数查询

为什么没有使用我的索引？（cont’）

- 情况5：此时如果用了索引，实际反而会更慢
- 情况6：没有正确的统计信息，造成CBO无法做出正确的选择
- 总结：归根到底，不使用索引的通常愿意就是“不能使用索引，使用索引会返回不正确的结果”，或者“不该使用索引，如果使用了索引就会变得更慢”

总结：索引访问的不同特点

- “查询使用了索引就万事大吉了”—误解啊～～
- 索引只是访问数据的一种方式
- “通过索引定位记录”只是查询工作的一部分
- 优化器有更多的选择权利
- 总结：索引不是万灵药。充分理解要处理的数据，做出合理的判断，才能获得高效方案

Practice in class 3-3

- 请研究你手上使用的数据库，比如，MySQL or Oracle，请研究数据库管理系统提供的其它索引形式，并阅读相关的文档
- 了解最常使用的3种索引的名称、结构、适用范围、不适用的范围、有益的例子、错误使用的例子等等，以及自己试一下。