CH8 Views & Indexes

一、视图

1. 声明

```
CREATE [MATERIALIZED] VIEW <view-name> AS <SQL query>;
```

- 虚拟，默认值，不会存储（仅仅会存储一个SELECT语句）
- 物化，实际创建和存储

2. 查询

same as normal query，但是会消耗更多时间

3. 删除

```
DROP VIEW <view-name>
```

4. 可更新的视图

无法通过视图修改数据。

可更新视图会将修改语句作用到基表上。

***本质：通过对视图的修改推导出背后基表的修改

特点：

- 朴素SELECT（没有聚合语句，DISTINCT等）
- WHERE中不能嵌套自己的子查询
- FROM中只能出现一个表

5. 触发器

可以将更新语句妥善翻译

```
CREATE TRIGGER PInsert
    INSTEAD OF INSERT ON PVIEW
    REFERENCING NEW ROW AS NewRow
    FOR EACH ROW
    INSERT INTO Movies(title, year, studioName)
    VALUES (NewRow.title, NewRow.year, 'Paramont');
```

6. 物化视图

对于使用频繁的视图可以创建物化视图来追求更高的查询效率

```
CREATE MATERIALIZED VIEW MovieProd AS ...
```

但是，基表改变时，每个物化视图都要随之更新。

增量保持性：只修改有变化的部分。

物化视图阶段性重构：通常在晚上。

二、索引

1. 声明

由“空间换时间”思想而来。

```
CREATE INDEX BeerIndex ON Beers(manf);
CREATE INDEX BeerIndex ON Beers(bar, beer);      -- 按照属性的顺序写
```

DBMS会在查询时自动选择是否使用索引。

2. 索引的选择

假设：

- 读写磁盘慢，读写内存块，不考虑Cache
- processor memory disk三层架构

SELECT： 读索引块→读数据块

INSERT/DELETE/UPDATE： 读索引块→读数据块→写回

***计算最佳索引：

填表。

共有p个pages

`SELECT a FROM Table WHERE b='111';`

平均一个b对应k个a，

键属性的k值为1s

则在有b的索引下，需要读1 + k次盘。

没有b的索引下，需要读p次盘。

如果属性c有*clustered*修饰，则WHERE条件中出现c的在有索引的情况下读2次，无索引的情况下读1次盘。

插入语句读盘次数为 （索引数 + 1）*2