



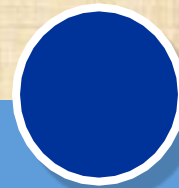
# GaussDB (for openGauss) ----- 创建和管理存储过程





# 索引

- ❖ 索引可以提高数据的访问速度，但同时也增加了插入、更新和删除表的处理时间。所以是否要为表增加索引，索引建立在哪些字段上，是创建索引前必须要考虑的问题。
- ❖ 需要分析应用程序的业务处理、数据使用、经常被用作查询条件或者被要求排序的字段来确定是否建立索引。
- ❖ openGauss支持4种创建索引的方式：唯一索引、多字段索引、部分索引、表达式索引。



# 创建索引

## ❖ 创建索引

在“美国各州县确诊与死亡数统计表”输入以下语句：

**CREATE INDEX 日期index ON 美国各州县确诊与死亡数统计表index (日期);**

The screenshot displays a database management tool interface. On the left, a sidebar shows the database structure with the following details:

- 库名: yiqing
- Schema: root
- Table list: 各国疫情数据统计表, 各国疫情数据统计表index, 病例基本信息表, 病例行程信息表, 美国各州县确诊与死亡数统计表 (expanded to show columns, indexes, and constraints), 美国各州县确诊与死亡数统计表1, 美国各州县确诊与死亡数统计表2, 美国各州县确诊与死亡数统计表3, 美国各州县确诊与死亡数统计表4.

The main area shows the SQL editor with the following content:

```
1 CREATE INDEX 日期index ON 美国各州县确诊与死亡数统计表 (日期);
2 |
```

Below the editor, the 'SQL执行记录' (SQL Execution Record) tab is active, showing the execution details:

-----开始执行-----

【拆分SQL完成】：将执行SQL语句数量：（1条）

【执行SQL：（1）】

CREATE INDEX 日期index ON 美国各州县确诊与死亡数统计表 (日期);

执行成功, 耗时: [151ms.]





# 管理索引

❖ **查询索引**：创建索引后刷新页面，左下角会显示表视图，单击indexes显示当前表的所有索引：

库名:  Schema:

表 视图

请按关键词搜索 |

- ▶ 各国疫情数据统计表
- ▶ 各国疫情数据统计表index
- ▶ 病例基本信息表
- ▶ 病例行程信息表
- ▼ 美国各州县确诊与死亡数统计表
  - ▶ columns
  - ▼ indexes
    - 日期index("日期")
  - ▶ constraints
- ▶ 美国各州县确诊与死亡数统计表1
- ▶ 美国各州县确诊与死亡数统计表2
- ▶ 美国各州县确诊与死亡数统计表3
- ▶ 美国各州县确诊与死亡数统计表4

我的SQL ▾

```
1 CREATE INDEX 日期index ON 美国各州县确诊与死亡数统计表 (日期);  
2 |
```

SQL执行记录 消息

-----开始执行-----

【拆分SQL完成】：将执行SQL语句数量：（1条）

【执行SQL：（1）】

CREATE INDEX 日期index ON 美国各州县确诊与死亡数统计表 (日期);  
执行成功,耗时: [151ms.]



# 删除索引

## ❖ 创建索引操作：

删除索引：DROP INDEX 日期index;





# 索引练习

❖ **索引练习**：对美国各州县确诊与死亡数统计表创建以下四类索引，尝试比较未建索引与创建索引后，查询效率的不同。

1) **创建唯一索引**：尝试比较未建索引后与创建索引后，查询效率的不同。

如果对于“美国各州县确诊与死亡数统计表1”，需要经常进行以下查询：

```
SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24'  
。
```

使用以下命令创建唯一索引：

```
CREATE INDEX 日期index1 ON 美国各州县确诊与死亡数统计表1 (日期);  
SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24';
```



# 索引练习

## ❖ 实验步骤 :

库名: yiqing Schema: root

SQL语句: `SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24';`

SQL执行记录

执行时间	SQL语句	消耗时间	执行结果
2021-11-15 06:28:47	SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24';	52 ms	执行成功
2021-11-15 06:28:47	set time zone 'PRC'	5 ms	执行成功
2021-11-15 06:28:35	SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24';	2 ms	执行失败
2021-11-15 06:28:34	set time zone 'PRC'	2 ms	执行成功
2021-11-15 06:28:25	SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24';	3 ms	执行失败

总条数: 100

库名: yiqing Schema: root

SQL语句: `SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24';`

SQL执行记录

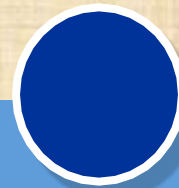
执行时间	SQL语句	消耗时间	执行结果
2021-11-15 06:47:24	SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24';	7 ms	执行成功
2021-11-15 06:47:23	set time zone 'PRC'	7 ms	执行成功
2021-11-15 06:46:59	SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24';	10 ms	执行成功
2021-11-15 06:46:58	set time zone 'PRC'	7 ms	执行成功
2021-11-15 06:46:45	SELECT 日期 FROM 美国各州县确诊与死亡数统计表1 WHERE 日期='2020-12-24';	3 ms	执行失败

总条数: 130





# 索引练习



## ❖ 实验步骤：

2) 创建多字段索引：尝试比较未建索引后与创建索引后，查询效率的不同。

若需要经常查询“美国各州县确诊与死亡数统计表2”中日期是‘2020-12-24’，且‘累计确诊’大于1000的记录，使用以下命令进行查询。

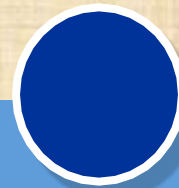
**SELECT \* FROM 美国各州县确诊与死亡数统计表2 WHERE 日期= '2020-12-24' AND 累计确诊 >1000;**使用以下命令在字段‘日期’和‘累计确诊’上定义一个多字段索引。

**CREATE INDEX 累计index ON 美国各州县确诊与死亡数统计表(日期 ,累计确诊);**





# 索引练习



## ❖ 实验步骤：

3) 创建部分索引：尝试比较未建索引后与创建索引后，查询效率的不同。

如果只需要查询日期='2020-12-24'的记录，可以创建部分索引来提升查询效率。

```
CREATE INDEX 日期index ON美国各州县确诊与死亡数统计表3(日期) WHERE 日期 = '2020-12-24';
```

4) 创建表达式索引：尝试比较未建索引后与创建索引后，查询效率的不同。

若经常需要查询'累计确诊'>1000的信息，执行如下命令进行查询。

```
SELECT * FROM 美国各州县确诊与死亡数统计表4 WHERE trunc(累计确诊) >1000;
```

可以为上面的查询创建表达式索引：CREATE INDEX 累计确诊\_index ON 美国各州县确诊与死亡数统计表4(trunc(累计确诊));



# 创建和管理视图

- ❖ 当用户对数据库中的一张或者多张表的某些字段的组合感兴趣，而又不想每次键入这些查询时，用户就可以定义一个视图，以便解决这个问题。
- ❖ 视图与基本表不同，不是物理上实际存在的，是一个虚表。数据库中仅存放视图的定义，而不存放视图对应的数据，这些数据仍存放在原来的基本表中。若基本表中的数据发生变化，从视图中查询出的数据也随之改变。从这个意义上讲，视图就像一个窗口，透过它可以看到数据库中用户感兴趣的数据及变化。视图每次被引用的时候都会运行一次。





# 创建和管理视图

- ❖ 执行如下命令创建普通视图bj\_yq : **CREATE VIEW bj\_yq**

执行SQL(F8) 格式化(F9) 执行计划(F6) 我的SQL ▾

```
1 CREATE VIEW bj_yq AS
2 SELECT 行程号,x.病例号,性别,x.日期信息,行程信息
3 FROM 病例行程信息表 AS x LEFT JOIN 病例基本信息表 AS y
4 ON x.病例号=y.病例号 WHERE y.省 = '北京市'
5
```

SQL执行记录 消息

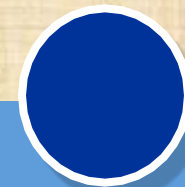
-----开始执行-----

【拆分SQL完成】：将执行SQL语句数量：（1条）

【执行SQL：（1）】

```
CREATE VIEW bj_yq123 AS
SELECT 行程号,x.病例号,性别,x.日期信息,行程信息
FROM 病例行程信息表 AS x LEFT JOIN 病例基本信息表 AS y
ON x.病例号=y.病例号 WHERE y.省 = '北京市'
执行成功，耗时：[14ms.]
```





# 创建和管理视图

- ❖ 执行如下命令创建普通视图bj\_yq : **CREATE VIEW bj\_yq**

执行SQL(F8) 格式化(F9) 执行计划(F6) 我的SQL ▾

```
1 CREATE VIEW bj_yq AS
2 SELECT 行程号,x.病例号,性别,x.日期信息,行程信息
3 FROM 病例行程信息表 AS x LEFT JOIN 病例基本信息表 AS y
4 ON x.病例号=y.病例号 WHERE y.省 = '北京市'
5
```

SQL执行记录 消息

-----开始执行-----

【拆分SQL完成】：将执行SQL语句数量：（1条）

【执行SQL：（1）】

```
CREATE VIEW bj_yq123 AS
SELECT 行程号,x.病例号,性别,x.日期信息,行程信息
FROM 病例行程信息表 AS x LEFT JOIN 病例基本信息表 AS y
ON x.病例号=y.病例号 WHERE y.省 = '北京市'
执行成功，耗时：[14ms.]
```



# 查询视图



当前所在库: **yiqing** [切换库](#) | 192.168.0.156,192.168.0.108,192.168.0.134:8000 | 字符集: UTF8

SQL窗口

- Schema列表
- 对象列表**
- 元数据采集

对象列表数据来自DAS后台定时采集任务, 最近一次采集时间: 2021-11-13 08:12:04 [立即采集](#) [清空采集数据](#)

表

视图

存储过程

触发器

序列

Schema: root

视图名称	操作
bj_yq	<div>打开视图   查看视图详情</div>

10 条/页

总条数: 1 < 1 >



# 查询视图



查看视图详情

## 查看视图详情



```
1 CREATE OR REPLACE VIEW "root"."bj_yq" as
2   SELECT x."行程号", x."病例号", y."性别", x."日期信息", x."行程信息" FROM (root."病例行程信息表"
```





# 查询视图



查看视图详情

查看视图详情

```
1 CREATE OR REPLACE VIEW "root"."bj_yq" as
2 SELECT x."行程号", x."病例号", y."性别", x."日期信息", x."行程信息" FROM (root."病例行程信息表"
```

- 1) 执行如下命令查询bj\_yq视图。SELECT  
\* FROM bj\_yq;



# 查询视图

查看视图详情

执行如下命令查询bj\_yq视图。**SELECT \* FROM bj\_yq;**





# 视图

## 实验步骤：

- ❖ 创建北京市病例信息的视图，包括行程号，病例号，性别，日期信息（选用病例行程信息表日期）和行程信息。
- ❖ 通过上述视图查询临床分型为普通型的病例号、行程号、性别和日期信息，按照病例号进行升序显示（截前五条记录）。





谢谢！！！！