的 PID。如果 pgid 是 0，那么就用 pid 指定的进程的 PID 作为进程组 ID。例如，如果进程 15213 是调用进程，那么

```
setpgid(0, 0);
```

会创建一个新的进程组，其进程组 ID 是 15213，并且把进程 15213 加入到这个新的进程组中。

2. 用/bin/kill 程序发送信号

/bin/kill 程序可以向另外的进程发送任意的信号。比如，命令

```
linux> /bin/kill -9 15213
```

发送信号 9(SIGKILL)给进程 15213。一个为负的 PID 会导致信号被发送到进程组 PID 中的每个进程。比如，命令

```
linux> /bin/kill -9 -15213
```

发送一个 SIGKILL 信号给进程组 15213 中的每个进程。注意，在此我们使用完整路径/bin/kill，因为有些 Unix shell 有自己内置的 kill 命令。

3. 从键盘发送信号

Unix shell 使用作业(job)这个抽象概念来表示为对一条命令行求值而创建的进程。在任何时刻，至多只有一个前台作业和 0 个或多个后台作业。比如，键入

```
linux> ls | sort
```

会创建一个由两个进程组成的前台作业，这两个进程是通过 Unix 管道连接起来的：一个进程运行 ls 程序，另一个运行 sort 程序。shell 为每个作业创建一个独立的进程组。进程组 ID 通常取自作业中父进程中的一个。比如，图 8-28 展示了有一个前台作业和两个后台作业的 shell。前台作业中的父进程 PID 为 20，进程组 ID 也为 20。父进程创建两个子进程，每个也都是进程组 20 的成员。

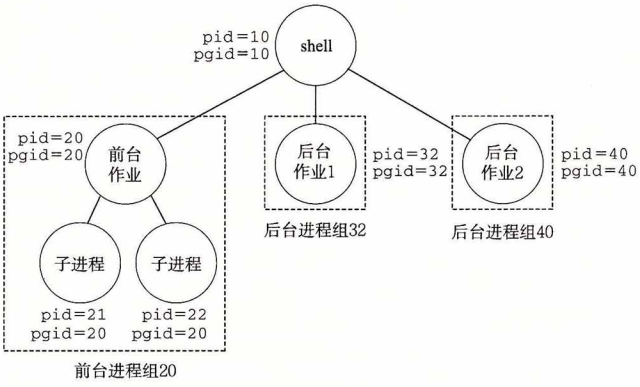

图 8-28　前台和后台进程组

在键盘上输入 Ctrl+C 会导致内核发送一个 SIGINT 信号到前台进程组中的每个进程。默认情况下，结果是终止前台作业。类似地，输入 Ctrl+Z 会发送一个 SIGTSTP 信号到前台进程组中的每个进程。默认情况下，结果是停止(挂起)前台作业。