随着数据量的增长，MySQL 已经满足不了大型互联网类应用的需求。因此，Redis 基于内存存储数据，可以极大的提高查询性能，对产品在架构上很好的补充。在某些场景下，可以充分的利用 Redis 的特性，大大提高效率。

### 缓存

对于热点数据，缓存以后可能读取数十万次，因此，对于热点数据，缓存的价值非常大。例如，分类栏目更新频率不高，但是绝大多数的页面都需要访问这个数据，因此读取频率相当高，可以考虑基于 Redis 实现缓存。

### 会话缓存

此外，还可以考虑使用 Redis 进行会话缓存。例如，将 web session 存放在 Redis 中。

### 时效性

例如验证码只有60秒有效期，超过时间无法使用，或者基于 Oauth2 的 Token 只能在 5 分钟内使用一次，超过时间也无法使用。

### 访问频率

出于减轻服务器的压力或防止恶意的洪水攻击的考虑，需要控制访问频率，例如限制 IP 在一段时间的最大访问量。

### 计数器

数据统计的需求非常普遍，通过原子递增保持计数。例如，应用数、资源数、点赞数、收藏数、分享数等。

### 社交列表

社交属性相关的列表信息，例如，用户点赞列表、用户分享列表、用户收藏列表、用户关注列表、用户粉丝列表等，使用 Hash 类型数据结构是个不错的选择。

### 记录用户判定信息

记录用户判定信息的需求也非常普遍，可以知道一个用户是否进行了某个操作。例如，用户是否点赞、用户是否收藏、用户是否分享等。

### 交集、并集和差集

在某些场景中，例如社交场景，通过交集、并集和差集运算，可以非常方便地实现共同好友，共同关注，共同偏好等社交关系。

### 热门列表与排行榜

按照得分进行排序，例如，展示最热、点击率最高、活跃度最高等条件的排名列表。

### 最新动态

按照时间顺序排列的最新动态，也是一个很好的应用，可以使用 Sorted Set 类型的分数权重存储 Unix 时间戳进行排序。

### 消息队列

Redis 能作为一个很好的消息队列来使用，依赖 List 类型利用 LPUSH 命令将数据添加到链表头部，通过 BRPOP 命令将元素从链表尾部取出。同时，市面上成熟的消息队列产品有很多，例如 RabbitMQ。因此，更加建议使用 RabbitMQ 作为消息中间件。