```c
// Name: Rodrigo Ignacio Rojas Garcia
// Course Number: ECE 2230
// Section: 001
// Semester: Spring 2017
// Assignment Number: 3
// Â© Rodrigo Rojas. All Rights Reserved.

// Library Declaration Section
#include <stdio.h>
#include <stdlib.h>
#include "disk_queue.h"
#include "list.h"
#include "structures.h"

// Typedef that defines structure disk_queue_s to have a pointers of same structure
 previous and next
struct disk_queue_s
{
   struct list *disk_queue;
};


// Function disk_queque_init allocates dynamic memory for a disk_queue_t by calling
 funciton list_init() from
// file list.c and will return the address of the allocated memory
disk_queue_t disk_queque_init(void)
{
    disk_queue_t disk_queue;
    disk_queue = (disk_queue_t)calloc(1, sizeof(struct disk_queue_s));
    disk_queue->disk_queue = list_init();
    return disk_queue;
}

// Function disk_queue_insert will call function list_append to insert a request at
 the end of databse disk_queue
int disk_queue_insert(disk_queue_t list, request_t request)
{
    int result;
    result = list_append(list->disk_queue, request);
    if (result == 0)
    {
        return 0;
    }
    else
    {
        return -1;
    }
}

// Function disk_queue_peek wil call function list_first to obtain the address of t
he first item on database disk_queue
// and if true it wil return item, if no item in database disk_queue it will return
 a NULL
request_t disk_queue_peek(disk_queue_t list)
{
    request_t first_request;
    first_request = list_first(list->disk_queue);
    if (first_request != NULL)
    {
        return first_request;
    }
    else
    {
        return NULL;
    }
}

// Function disk_queue_peek will call function list_remove to remove current reques
t on database disk_queue and will return the
// address of current if succesfull, if not it returns a NULL
request_t disk_queue_remove(disk_queue_t list)
{
    request_t first_request;
    first_request = list_first(list->disk_queue);
    if (first_request != NULL)
    {
        list_remove(list->disk_queue);
        return first_request;
    }
    else
    {
        return NULL;
    }
}

// Function will check if there is a request on the disk_queue database and if true
 it will return 0 to signify that, this is true, if not it returns
// -1 to signify false
int disk_queue_empty(disk_queue_t list)
{
    request_t first_request;
    first_request = list_first(list->disk_queue);
    if (first_request != NULL)
    {
        return 0;
    }
    else
    {
        return -1;
    }
}

// Function disk_queue_finalize will call function list_finalize which will free ea
ch dynamic memory called for each single database in disk_queue
void disk_queue_finalize(disk_queue_t list)
{
    list_finalize(list->disk_queue);
    free(list);
}
```