**Grupo: 1202**

Andrés Salas Peña y Rodrigo Alonso de Pool Alcántara

**Funcionalidades:**

|  |  |
| --- | --- |
| **Funcionalidad** | Implementado (SI / NO) |
| **Tipos de datos** | |
| DBL | SI |
| LNG | SI |
| **Módulos** | |
| table.c | SI |
| record.c | SI |
| **Operaciones** | |
| COUNT | SI |
| UNION | SI |
| LIMIT | SI |
| OFFSET | SI |

**Lista de libros comprados por "jack":**

|  |  |
| --- | --- |
| ESECUELE | SQL |
| usuarios SEQUENTIAL 1 STR Jack C\_COLEQCTE SELECT INT 0 P\_COL 1 PROJECT ventas SEQUENTIAL PRODUCT 0 2 C\_COLEQCOL SELECT INT 1 P\_COL 1 PROJECT edicion SEQUENTIAL PRODUCT 0 1 C\_COLEQCOL SELECT INT 0 P\_COL STR 2 P\_COL 2 PROJECT | Select edicion\_id, titulo  From Edicion NATURAL JOIN Ventas NATURAL JOIN Usuarios  Where nombre = ‘Jack’ |

**Números de libros comprados por “jack”:**

|  |  |
| --- | --- |
| ESECUELE | SQL |
| usuarios SEQUENTIAL 1 STR Jack C\_COLEQCTE SELECT INT 0 P\_COL 1 PROJECT ventas SEQUENTIAL PRODUCT 0 2 C\_COLEQCOL SELECT INT 1 P\_COL 1 PROJECT edicion SEQUENTIAL PRODUCT 0 1 C\_COLEQCOL SELECT INT 0 P\_COL STR 2 P\_COL 2 PROJECT COUNT | Select count(\*)  From Ventas NATURAL JOIN Usuarios  Where nombre = ‘Jack’ |

**ER de las tablas** ***edición***, ***ventas*** y ***usuarios***:

|  |  |  |
| --- | --- | --- |
| EDICION | | |
| edicion\_id | titulo | cantidad |

|  |  |
| --- | --- |
| VENTAS | |
| edicion\_id | usuario\_id |

|  |  |
| --- | --- |
| USUARIOS | |
| usuario\_id  N  VENTAS  M  USUARIOS  EDICION | nombre |

**Ejemplos del funcionamiento:**

Se debe entregar un ejemplo de funcionamiento de cada una de las funciones implementadas para la base de datos “*libros\_db*”. Para ello se puede reutilizar el “.bash” que se entrega con la práctica.

Para observar el correcto funcionamiento de nuestras operaciones se adjunta el bank.bash con las nuevas consultas realizadas, de todas formas se puede observar cuales son las consultas realizadas para probar las operaciones aquí abajo:

# Probando la operacion COUNT

# - Count the number of clients

$COMMAND query bank\_db << EOF

clients SEQUENTIAL COUNT

EOF

# Probando la operacion UNION

# - Do the union of the same table accounts

$COMMAND query bank\_db << EOF

accounts SEQUENTIAL accounts SEQUENTIAL UNION

EOF

# Probando la operacion LIMIT

# -Show the first five rows of table accounts

$COMMAND query bank\_db << EOF

accounts SEQUENTIAL 5 LIMIT

EOF

# Probando la operacion OFFSET

## -Show the rows from position 5 of table accounts

$COMMAND query bank\_db << EOF

accounts SEQUENTIAl 5 OFFSET

EOF

# Probando todas las operaciones

## -Count the number of first five accounts union the rows from position 5 of table accounts.

$COMMAND query bank\_db << EOF

accounts SEQUENTIAL 5 OFFSET accounts SEQUENTIAL 5 LIMIT UNION COUNT

EOF

NOTA:

Nuestros scripts son directamente creados tras el comando make esecuele, en la carpeta install. Los scripts que se encontrará son:

- bank.bash y un fichero .bash por cada subacción ejecutada. De este modo puede probar cada acción por separado.

- libros.bash y ,al igual que en bank.bash, un fichero .bash por cada subacción ejecutada.

Para ejecutar el main de prueba de nuestras tables y records se debe ejecutar el makefile que se encuentra en la carpeta development/database/ y tras hacer el make ejecutar el fichero test que se genera.