Week03 - Testing Database

SQL to check the tables

**Q1. Check that each table will display the output presented in the individual tables**

|  |  |  |
| --- | --- | --- |
| Code | Expected Output (Comment) | Actual Output (Screenshot) |
| select \* from customer; | All records display (10 records) |  |
| select \* from staff; | All records display (10 records) |  |
| select \* from menu; | All records display  (5 records) |  |
| select \* from outlet; | All records display (5 records) |  |
| select \* from payment; | All records display (30 records) |  |
| select \* from item; | All records display (20 records) |  |
| select \* from ordering; | All records display (30 records) |  |
| select \* from item\_order; | All records display (56 records -  screenshot can only show 44 records) |  |

**Q2. Show the output from two of the adjacent tables in turn – that is orders and item, users and item…**

|  |  |  |
| --- | --- | --- |
| Code | Expected Output (Comment) | Actual Output (Screenshot) |
| select \* from ordering, payment where ordering.payment\_id = payment.payment\_id  OR  select \* from ordering **join** payment  **on**  ordering.payment\_id = payment.payment\_id | All columns from **ordering** and **payment** tables; 30 records as there is a 1-1 relation between the two tables. |  |
| select \* from customer, ordering where customer.customer\_id = ordering.customer\_id order by customer.customer\_id;  OR  select \* from customer **join** ordering **on** customer.customer\_id = ordering.customer\_id order by customer.customer\_id; | For each customer, the orders of that customer. All the columns of each table. Customer’s records are repeated as many times as the orders he has placed. Number of records is the number of orders (30). |  |
| select \* from menu, item where menu.menu\_no = item.menu\_no order by menu.menu\_no;  OR select \* from menu **join** item **on** menu.menu\_no = item.menu\_no order by menu.menu\_no; | For each menu, the items included in that menu. All the columns of each table. Menu columns are repeated for as many times as the items in that menu. There are 20 records, as the number of items. |  |
| select \* from staff, ordering where staff.staff\_id = ordering.staff\_id order by staff.staff\_id;  OR select \* from staff **join** ordering **on** staff.staff\_id = ordering.staff\_id order by staff.staff\_id; | For each staff, the orders taken by that staff. All the columns of each table. Staff records are repeated as many times as the orders he has taken. Number of records is the number of orders (30). |  |
| select \* from customer, payment where customer.customer\_id = payment.customer\_id order by customer.customer\_id;  OR  select \* from customer **join** payment **on** customer.customer\_id = payment.customer\_id order by customer.customer\_id; | For each customer, the payments of that customer. All the columns of each table. Customer’s records are repeated as many times as the payments he has executed. Number of records is the number of payments (30). |  |

**Q3. Connect all three tables and display the output that shows the output from these three tables – customer, ordering and item.**

|  |  |  |
| --- | --- | --- |
| Code | Expected Output (Comment) | Actual Output (Screenshot) |
| select \* from customer, ordering, item\_order, item where customer.customer\_id = ordering.customer\_id and ordering.order\_id = item\_order.order\_id and item\_order.item\_id = item.item\_id order by customer.customer\_id, ordering.order\_id; | For each customer, the orders he made and for each order, the items included in the order. The records are more than the number of orders (56 vs 30) because the record of one order is repeated for each item included in the order.  Please note not all columns and records could be included in the screeenshot. |  |

Using joins, the above query could be written as:

select \*

from

customer

**join** ordering **on** customer.customer\_id = ordering.customer\_id

**join** item\_order **on** ordering.order\_id = item\_order.order\_id

**join** item **on** item\_order.item\_id = item.item\_id

order by customer.customer\_id, ordering.order\_id;

**Q4. Create a query that will show the customer and items – so link these two**

|  |  |  |
| --- | --- | --- |
| Code | Expected Output | Actual Output (Screenshot) |
| select customer.customer\_id, customer.customer\_firstname, customer.customer\_surname, item.item\_name, item.item\_type, ordering.order\_id from customer, ordering, item\_order, item where customer.customer\_id = ordering.customer\_id and ordering.order\_id = item\_order.order\_id and item\_order.item\_id = item.item\_id order by customer.customer\_id, ordering.order\_id; | For each customer, his/her name and surname, the id of the orders he made and for each order, the name and type of the items included in the order. The records are more than the number of orders (56 vs 30) because the record of one order is repeated for each item included in the order.  Please note not all records could be included in the screeenshot. |  |

**Q5. A manager wants to show a catalog of the items in the system – but only the following attributes showing: customer\_id, customer\_name, item\_price**

First I need to add the ‘item\_price’ column to the item table:

ALTER TABLE item ADD COLUMN item\_price DECIMAL(6,2) NOT NULL;

Then I truncate the table:

TRUNCATE TABLE item;

And reinsert the data with also the price.

This is the new item table:

|  |  |  |
| --- | --- | --- |
| Code | Expected Output (Comment) | Actual Output (Screenshot) |
| select \* from item; | All records (20 items) |  |

Now I can query the database to only get the information requested.

|  |  |  |
| --- | --- | --- |
| Code | Expected Output | Actual Output (Screenshot) |
| select customer.customer\_id, customer.customer\_firstname, customer.customer\_surname, item.item\_name, item.item\_price from customer, ordering, item\_order, item where customer.customer\_id = ordering.customer\_id and ordering.order\_id = item\_order.order\_id and item\_order.item\_id = item.item\_id order by customer.customer\_id, ordering.order\_id;  OR  select customer.customer\_id, customer.customer\_firstname, customer.customer\_surname, item.item\_name, item.item\_price from customer **join** ordering **on** customer.customer\_id = ordering.customer\_id **join** item\_order **on** ordering.order\_id = item\_order.order\_id **join** item **on** item\_order.item\_id = item.item\_id order by customer.customer\_id, ordering.order\_id; | For each customer, his/her name and surname, the name and price of the items he ordered (over many orders). The records are more than the number of orders (56 vs 30) because the record of one order is repeated for each item included in the order.  Please note not all records could be included in the screeenshot. |  |