

# PRÁCTICA SQL Y DW

## 1. DIAGRAMA ENTIDAD RELACIÓN

Realizar el diagrama entidad relación con el que poder modelar una base de datos para Keepcoding, es decir, recogiendo datos de alumnos, bootcamps, módulos, profesores...

Se debe entregar un pdf con el diagrama y una pequeña explicación de este.

## 2. CREACIÓN DE BASE DE DATOS

Desarrollar un script para crear las tablas y las restricciones necesarias según el diagrama entregado en el punto anterior.

El script debe poder ejecutarse en PostgreSQL.

Se debe entregar un fichero con el código solicitado y extensión .sql

## 3. CREAR TABLA DE ivr\_detail.

Vamos a realizar el modelo de datos correspondiente a una [IVR](#) de atención al cliente.

Desde los ficheros ivr\_calls, ivr\_modules, e ivr\_steps crear las tablas con los mismos nombres dentro del dataset keepcoding.

En ivr\_calls encontramos los datos referentes a las llamadas.

En ivr\_modules encontramos los datos correspondientes a los diferentes módulos por los que pasa la llamada. Se relaciona con la tabla de ivr\_calls a través del campo ivr\_id.

En ivr\_steps encontramos los datos correspondientes a los pasos que da el usuario dentro de un módulo. Se relaciona con la tabla de módulos a través de los campos ivr\_id y module\_sequence.

Queremos tener los siguientes campos:

calls\_ivr\_id

calls\_phone\_number

calls\_ivr\_result

calls\_vdn\_label

calls\_start\_date

calls\_start\_date\_id

calls\_end\_date

calls\_end\_date\_id

calls\_total\_duration

calls\_customer\_segment

calls\_ivr\_language

calls\_steps\_module

```
calls_module_aggregation
module_sequece
module_name
module_duration
module_result
step_sequence
step_name
step_result
step_description_error
document_type
document_identification
customer_phone
billing_account_id
```

Los campos `calls_start_date_id` y `calls_end_date_id` son campos de fecha calculados, del tipo `yyyymmdd`. Por ejemplo, el 1 de enero de 2023 sería `20230101`.

Entregar el código SQL que generaría la tabla `ivr_detail` en un fichero `.sql`. La tabla se debe crear dentro del dataset `keepcoding`, en la query no debe aparecer el nombre del proyecto de `gcp`.

#### 4. Generar el campo `vdn_aggregation`

Generar el campo para cada llamada, es decir, queremos tener el campo `calls_ivr_id` y el campo `vdn_aggregation` con la siguiente lógica:

es una generalización del campo `vdn_label`. Si `vdn_label` empieza por `ATC` pondremos `FRONT`, si empieza por `TECH` pondremos `TECH` si es `ABSORPTION` dejaremos `ABSORPTION` y si no es ninguna de las anteriores pondremos `RESTO`.

Entregar el código en un fichero `.sql`

#### 5. Generar los campos `document_type` y `document_identification`

En ocasiones es posible identificar al cliente en alguno de los pasos de detail obteniendo su tipo de documento y su identificación.

Como en el ejercicio anterior queremos tener un registro por cada llamada y un sólo cliente identificado para la misma.

Entregar el código en un fichero .sql

## 6. Generar el campo customer\_phone

En ocasiones es posible identificar al cliente en alguno de los pasos de detail obteniendo su número de teléfono.

Como en el ejercicio anterior queremos tener un registro por cada llamada y un sólo cliente identificado para la misma.

Entregar el código en un fichero .sql

## 7. Generar el campo billing\_account\_id

En ocasiones es posible identificar al cliente en alguno de los pasos de detail obteniendo su número de teléfono.

Como en el ejercicio anterior queremos tener un registro por cada llamada y un sólo cliente identificado para la misma.

Entregar el código en un fichero .sql

## 8. Generar el campo masiva\_lg

Como en el ejercicio anterior queremos tener un registro por cada llamada y un flag que indique si la llamada ha pasado por el módulo AVERIA\_MASIVA. Si es así indicarlo con un 1 de lo contrario con un 0.

Entregar el código en un fichero .sql

## 9. Generar el campo info\_by\_phone\_lg

Como en el ejercicio anterior queremos tener un registro por cada llamada y un flag que indique si la llamada pasa por el step de nombre CUSTOMERINFOBYPHONE.TX y su step\_result es OK, quiere decir que hemos podido identificar al cliente a través de su número de teléfono. En ese caso pondremos un 1 en este flag, de lo contrario llevará un 0.

Entregar el código en un fichero .sql

## 10. Generar el campo info\_by\_dni\_lg

Como en el ejercicio anterior queremos tener un registro por cada llamada y un flag que indique si la llamada pasa por el step de nombre CUSTOMERINFOBYDNI.TX y su step\_result es OK,, quiere decir que hemos podido identificar al cliente a través de su número de dni. En ese caso pondremos un 1 en este flag, de lo contrario llevará un 0.

Entregar el código en un fichero .sql

## 11. Generar los campos repeated\_phone\_24H, cause\_recall\_phone\_24H

Como en el ejercicio anterior queremos tener un registro por cada llamada y dos flags que indiquen si el calls\_phone\_number tiene una llamada las anteriores 24 horas o en las siguientes 24 horas. En caso afirmativo pondremos un 1 en estos flag, de lo contrario llevará un 0.

Entregar el código en un fichero .sql

## 12. CREAR TABLA DE ivr\_summary (Para nota)

Con la base de la tabla ivr\_detail y el código de todos los ejercicios anteriores vamos a crear la tabla ivr\_summary . Ésta será un resumen de la llamada donde se incluyen los indicadores más importantes de la llamada. Por tanto, sólo tendrá un registro por llamada.

Queremos que tengan los siguientes campos:

**ivr\_id:** identificador de la llamada (viene de detail).  
**phone\_number:** número llamante (viene de detail).  
**ivr\_result:** resultado de la llamada (viene de detail).  
**vdn\_aggregation:** calculado anteriormente.  
**start\_date:** fecha inicio de la llamada (viene de detail).  
**end\_date:** fecha fin de la llamada (viene de detail).  
**total\_duration:** duración de la llamada (viene de detail).  
**customer\_segment:** segmento del cliente (viene de detail).  
**ivr\_language:** idioma de la IVR (viene de detail).

**steps\_module:** número de módulos por los que pasa la llamada (viene de detail).

**module\_aggregation:** lista de módulos por los que pasa la llamada (viene de detail).

**document\_type:** calculado anteriormente.

**document\_identification:** calculado anteriormente.

**customer\_phone:** calculado anteriormente.

**billing\_account\_id:** calculado anteriormente.

**masiva\_lg:** calculado anteriormente.

**info\_by\_phone\_lg:** calculado anteriormente.

**info\_by\_dni\_lg:** calculado anteriormente.

**repeated\_phone\_24H:** calculado anteriormente.

**cause\_recall\_phone\_24H:** calculado anteriormente.

Entregar el código SQL que generaría la tabla `ivr_summary` dentro del dataset `keepcoding`.

## 13. CREAR FUNCIÓN DE LIMPIEZA DE ENTEROS

Crear una función de limpieza de enteros por la que si entra un null la función devuelva el valor -999999.

Entregar el código SQL que generaría la función `clean_integer` dentro del dataset `keepcoding`.