PRIMERA PARTE

Estudio de STG_PRODUCTOS_CRM (I)

```sql
USE STAGE;

SELECT COUNT(*) TOTAL_REGISTROS,
SUM(CASE WHEN LENGTH(TRIM(PRODUCTO_ID)) <> 0 THEN 1 ELSE 0 END) TOTAL_PRODUCTO_ID,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(PRODUCTO_ID)) <> 0 THEN PRODUCTO_ID ELSE 0 END) TOTAL_DISNTINTOS_PRODUCTO_ID,
SUM(CASE WHEN LENGTH(TRIM(CUSTOMER_ID)) <> 0 THEN 1 ELSE 0 END) TOTAL_CUSTOMER_ID,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(CUSTOMER_ID)) <> 0 THEN CUSTOMER_ID ELSE 0 END) TOTAL_DISNTINTOS_CUSTOMER_ID,
SUM(CASE WHEN LENGTH(TRIM(PRODUCT_NAME)) <> 0 THEN 1 ELSE 0 END) TOTAL_PRODUCT_NAME,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(PRODUCT_NAME)) <> 0 THEN PRODUCT_NAME ELSE 0 END) TOTAL_DISNTINTOS_PRODUCT_NAME,
SUM(CASE WHEN LENGTH(TRIM(ACCESS_POINT)) <> 0 THEN 1 ELSE 0 END) TOTAL_ACCESS_POINT,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(ACCESS_POINT)) <> 0 THEN ACCESS_POINT ELSE 0 END) TOTAL_DISNTINTOS_ACCESS_POINT,
SUM(CASE WHEN LENGTH(TRIM(CHANNEL)) <> 0 THEN 1 ELSE 0 END) TOTAL_CHANNEL,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(CHANNEL)) <> 0 THEN CHANNEL ELSE 0 END) TOTAL_DISNTINTOS_CHANNEL,
SUM(CASE WHEN LENGTH(TRIM(AGENT_CODE)) <> 0 THEN 1 ELSE 0 END) TOTAL_AGENT_CODE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(AGENT_CODE)) <> 0 THEN AGENT_CODE ELSE 0 END) TOTAL_DISNTINTOS_AGENT_CODE,
SUM(CASE WHEN LENGTH(TRIM(START_DATE)) <> 0 THEN 1 ELSE 0 END) TOTAL_START_DATE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(START_DATE)) <> 0 THEN START_DATE ELSE 0 END) TOTAL_DISNTINTOS_START_DATE,
SUM(CASE WHEN LENGTH(TRIM(INSTALL_DATE)) <> 0 THEN 1 ELSE 0 END) TOTAL_INSTALL_DATE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(INSTALL_DATE)) <> 0 THEN INSTALL_DATE ELSE 0 END) TOTAL_DISNTINTOS_INSTALL_DATE,
SUM(CASE WHEN LENGTH(TRIM(END_DATE)) <> 0 THEN 1 ELSE 0 END) TOTAL_END_DATE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(END_DATE)) <> 0 THEN END_DATE ELSE 0 END) TOTAL_DISNTINTOS_END_DATE,
SUM(CASE WHEN LENGTH(TRIM(PRODUCT_CITY)) <> 0 THEN 1 ELSE 0 END) TOTAL_PRODUCT_CITY,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(PRODUCT_CITY)) <> 0 THEN PRODUCT_CITY ELSE 0 END) TOTAL_DISNTINTOS_PRODUCT_CITY,
SUM(CASE WHEN LENGTH(TRIM(PRODUCT_ADDRESS)) <> 0 THEN 1 ELSE 0 END) TOTAL_PRODUCT_ADDRESS,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(PRODUCT_ADDRESS)) <> 0 THEN PRODUCT_ADDRESS ELSE 0 END) TOTAL_DISNTINTOS_PRODUCT_ADDRESS,
SUM(CASE WHEN LENGTH(TRIM(PRODUCT_POSTAL_CODE)) <> 0 THEN 1 ELSE 0 END) TOTAL_PRODUCT_POSTAL_CODE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(PRODUCT_POSTAL_CODE)) <> 0 THEN PRODUCT_POSTAL_CODE ELSE 0 END) TOTAL_DISNTINTOS_PRODUCT_POSTAL_CODE,
SUM(CASE WHEN LENGTH(TRIM(PRODUCT_STATE)) <> 0 THEN 1 ELSE 0 END) TOTAL_PRODUCT_STATE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(PRODUCT_STATE)) <> 0 THEN PRODUCT_STATE ELSE 0 END) TOTAL_DISNTINTOS_PRODUCT_STATE,
SUM(CASE WHEN LENGTH(TRIM(PRODUCT_COUNTRY)) <> 0 THEN 1 ELSE 0 END) TOTAL_PRODUCT_COUNTRY,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(PRODUCT_COUNTRY)) <> 0 THEN PRODUCT_COUNTRY ELSE 0 END) TOTAL_DISNTINTOS_PRODUCT_COUNTRY
FROM STG_PRODUCTOS_CRM;
```

Estudio de STG_PRODUCTOS_CRM (II)

TOTAL_REGISTROS:	78495
TOTAL_PRODUCTO_ID:	78495
TOTAL_DISNTINTOS_PRODUCTO_ID:	78495
TOTAL_CUSTOMER_ID:	78495
TOTAL_DISNTINTOS_CUSTOMER_ID:	8001
TOTAL_PRODUCT_NAME:	78495
TOTAL_DISNTINTOS_PRODUCT_NAME:	6
TOTAL_ACCESS_POINT:	78274
TOTAL_DISNTINTOS_ACCESS_POINT:	78275
TOTAL_CHANNEL:	78274
TOTAL_DISNTINTOS_CHANNEL:	5
TOTAL_AGENT_CODE:	42630
TOTAL_DISNTINTOS_AGENT_CODE:	701
TOTAL_START_DATE:	78495
TOTAL_DISNTINTOS_START_DATE:	8035
TOTAL_INSTALL_DATE:	75363
TOTAL_DISNTINTOS_INSTALL_DATE:	75360

TOTAL_END_DATE:	46684
TOTAL_DISNTINTOS_END_DATE:	46683
TOTAL_PRODUCT_CITY:	78274
TOTAL_DISNTINTOS_PRODUCT_CITY:	82
TOTAL_PRODUCT_ADDRESS:	78274
TOTAL_DISNTINTOS_PRODUCT_ADDRESS:	77037
TOTAL_PRODUCT_POSTAL_CODE:	78274
TOTAL_DISNTINTOS_PRODUCT_POSTAL_CODE:	274
TOTAL_PRODUCT_STATE:	78090
TOTAL_DISNTINTOS_PRODUCT_STATE:	4
TOTAL_PRODUCT_COUNTRY:	78274
TOTAL_DISNTINTOS_PRODUCT_COUNTRY:	2

Estudio de STG_PRODUCTOS_CRM (III)

Conclusiones:

- Si observamos los campos de la tabla STG_PRODUCTOS_CRM, vemos que en realidad no se están almacenando productos en sí, sino los datos de los diferentes SERVICIOS contratados por los clientes de KC Telecom S.L.

- Dado que el campo PRODUCTO_ID no se repite en ningún registro, y además, no presenta valores nulos, será el que usaremos como Clave Principal de la tabla de ODS_HC_SERVICIOS

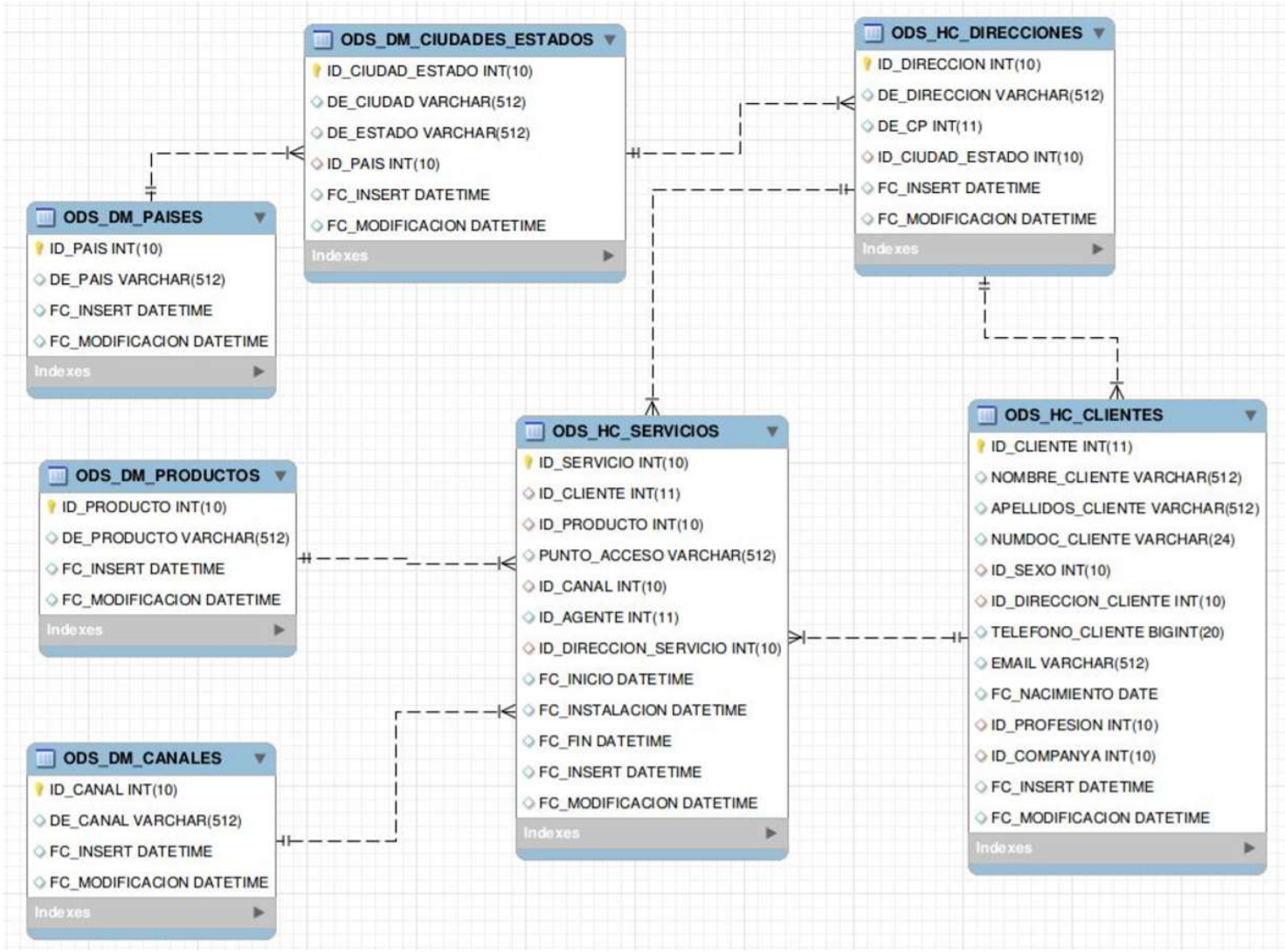
- El campo CUSTOMER_ID será la Clave Ajena respecto la tabla ODS_HC_CLIENTES
  - Hay tres registros cuyos clientes no se encuentran en ODS_HC_CLIENTE

- Dado que sólo hay 5 canales diferentes, deberíamos crear tabla dimensional para almacenarlos

- También crearemos una tabla dimensional para los AGENT_CODE
  - Hay 35865 registros con AGENT_CODE en blanco

- Añadiremos las direcciones de las diferentes instalaciones de suministros, y las añadiremos a la tabla ODS_HC_DIRECCIONES
  - En esta tabla se ha establecido la cadena ‘United States’ en lugar de ‘US’
  - Hay direcciones incompletas, que también almacenaremos para guardar la trazabilidad de las mismas

Creamos modelo de SERVICIOS



Creamos las tablas de SERVICIOS

```
USE ODS;

DROP TABLE IF EXISTS ODS_DM_PRODUCTOS;
CREATE TABLE IF NOT EXISTS ODS_DM_PRODUCTOS (
        ID_PRODUCTO INT UNSIGNED AUTO_INCREMENT NOT NULL PRIMARY KEY,
        DE_PRODUCTO VARCHAR(512),
        FC_INSERT DATETIME,
        FC_MODIFICACION DATETIME);

DROP TABLE IF EXISTS ODS_DM_CANALES;
CREATE TABLE IF NOT EXISTS ODS_DM_CANALES (
        ID_CANAL INT UNSIGNED AUTO_INCREMENT NOT NULL PRIMARY KEY,
        DE_CANAL VARCHAR(512),
        FC_INSERT DATETIME,
        FC_MODIFICACION DATETIME);

DROP TABLE IF EXISTS ODS_HC_SERVICIOS;
CREATE TABLE IF NOT EXISTS ODS_HC_SERVICIOS (
        ID_SERVICIO INT(11) UNSIGNED NOT NULL PRIMARY KEY,
        ID_CLIENTE INT(11),
        FLG_CLIENTE_EXISTENTE TINYINT(1)
        ID_PRODUCTO INT,
        PUNTO_ACCESO VARCHAR(512),
        ID_CANAL INT,
        ID_AGENTE INT(11),
        ID_DIRECCION_SERVICIO INT,
        FC_INICIO DATETIME,
        FC_INSTALACION DATETIME,
        FC_FIN DATETIME,
        FC_INSERT DATETIME,
    FC_MODIFICACION DATETIME);
```

Creamos las FK del modelo de SERVICIOS

```
USE ODS;

ALTER TABLE ODS_HC_SERVICIOS MODIFY COLUMN ID_PRODUCTO INT(10) UNSIGNED;
ALTER TABLE ODS_HC_SERVICIOS ADD INDEX fk_serv_prod_idx (ID_PRODUCTO ASC);
ALTER TABLE ODS_HC_SERVICIOS ADD CONSTRAINT fk_serv_prod FOREIGN KEY (ID_PRODUCTO)
    REFERENCES ODS_DM_PRODUCTOS (ID_PRODUCTO);

ALTER TABLE ODS_HC_SERVICIOS MODIFY COLUMN ID_CANAL INT(10) UNSIGNED;
ALTER TABLE ODS_HC_SERVICIOS ADD INDEX fk_serv_can_idx (ID_CANAL ASC);
ALTER TABLE ODS_HC_SERVICIOS ADD CONSTRAINT fk_serv_can FOREIGN KEY (ID_CANAL)
    REFERENCES ODS_DM_CANALES (ID_CANAL);

ALTER TABLE ODS_HC_SERVICIOS MODIFY COLUMN ID_DIRECCION_SERVICIO INT(10) UNSIGNED;
ALTER TABLE ODS_HC_SERVICIOS ADD INDEX fk_serv_dir_idx (ID_DIRECCION_SERVICIO ASC);
ALTER TABLE ODS_HC_SERVICIOS ADD CONSTRAINT fk_serv_dir FOREIGN KEY (ID_DIRECCION_SERVICIO)
    REFERENCES ODS_HC_DIRECCIONES (ID_DIRECCION);

ALTER TABLE ODS_HC_SERVICIOS MODIFY COLUMN ID_CLIENTE INT(11);
ALTER TABLE ODS_HC_SERVICIOS ADD INDEX fk_serv_cli_idx (ID_CLIENTE ASC);
ALTER TABLE ODS_HC_SERVICIOS ADD CONSTRAINT fk_serv_cli FOREIGN KEY (ID_CLIENTE)
REFERENCES ODS_HC_CLIENTES (ID_CLIENTE);
```

Poblamos el modelo de SERVICIOS (I)

```
USE ODS;

INSERT INTO ODS_DM_PRODUCTOS (DE_PRODUCTO, FC_INSERT, FC_MODIFICACION)
    SELECT DISTINCT UPPER(TRIM(PRODUCT_NAME)), NOW(), NOW()
    FROM STAGE.STG_PRODUCTOS_CRM
    WHERE TRIM(PRODUCT_NAME) <> '';
COMMIT;
INSERT INTO ODS_DM_PRODUCTOS VALUES (99, 'DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS_DM_PRODUCTOS VALUES (98, 'NO APLICA', NOW(), NOW());
COMMIT;

INSERT INTO ODS_DM_CANALES (DE_CANAL, FC_INSERT, FC_MODIFICACION)
    SELECT DISTINCT UPPER(TRIM(CHANNEL)), NOW(), NOW()
    FROM STAGE.STG_PRODUCTOS_CRM
    WHERE TRIM(CHANNEL) <> '';
COMMIT;
INSERT INTO ODS_DM_CANALES VALUES (99, 'DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS_DM_CANALES VALUES (98, 'NO APLICA', NOW(), NOW());
COMMIT;
```

Poblamos el modelo de SERVICIOS (II)

```
USE ODS;

INSERT INTO ODS_DM_PAISES (DE_PAIS, FC_INSERT, FC_MODIFICACION)
SELECT DISTINCT UPPER(TRIM(PRODUCT_COUNTRY))
FROM STAGE.STG_PRODUCTOS_CRM PROD
WHERE TRIM(PRODUCT_COUNTRY) <>'' AND
TRIM(REPLACE(PROD.PRODUCT_COUNTRY, 'United States', 'US')) NOT IN
(SELECT DE_PAIS
FROM ODS.ODS_DM_PAISES);
COMMIT;
ANALYZE TABLE ODS_DM_PAISES;

INSERT INTO ODS_DM_CIUDADES_ESTADOS (DE_CIUDAD, DE_ESTADO, ID_PAIS, FC_INSERT, FC_MODIFICACION)
SELECT DISTINCT
CASE WHEN LENGTH(PRODUCT_CITY) <> 0 THEN UPPER(TRIM(PRODUCT_CITY)) ELSE 'DESCONOCIDO' END,
CASE WHEN LENGTH(PRODUCT_STATE) <> 0 THEN UPPER(TRIM(PRODUCT_STATE)) ELSE 'DESCONOCIDO' END,
PAI.ID_PAIS,
NOW(),
NOW()
FROM STAGE.STG_PRODUCTOS_CRM PROD
INNER JOIN ODS.ODS_DM_PAISES PAI ON CASE WHEN TRIM(PRODUCT_COUNTRY) <> '' THEN
        UPPER(REPLACE(PRODUCT_COUNTRY, 'United States', 'US'))
    ELSE
        'DESCONOCIDO'
    END = PAI.DE_PAIS
WHERE NOT EXISTS
(SELECT 1
FROM ODS.ODS_DM_CIUDADES_ESTADOS CIU
WHERE CASE WHEN LENGTH(PRODUCT_CITY) <> 0 THEN UPPER(TRIM(PRODUCT_CITY)) ELSE 'DESCONOCIDO' END = CIU.DE_CIUDAD AND
CASE WHEN LENGTH(PRODUCT_STATE) <> 0 THEN UPPER(TRIM(PRODUCT_STATE)) ELSE 'DESCONOCIDO' END = CIU.DE_ESTADO AND
PAI.ID_PAIS = CIU.ID_PAIS);
COMMIT;
ANALYZE TABLE ODS_DM_CIUDADES_ESTADOS;
```

Poblamos el modelo de SERVICIOS (III)

```
INSERT INTO ODS_HC_DIRECCIONES (DE_DIRECCION, DE_CP, ID_CIUDAD_ESTADO, FC_INSERT, FC_MODIFICACION)
    SELECT DISTINCT
        CASE WHEN TRIM(PRODUCT_ADDRESS) <> '' THEN UPPER(TRIM(PRODUCT_ADDRESS)) ELSE 'DESCONOCIDO' END DIRECCION,
        CASE WHEN LENGTH(TRIM(PRODUCT_POSTAL_CODE)) <> 0 THEN TRIM(PRODUCT_POSTAL_CODE) ELSE 99999 END CP,
        CIU.ID_CIUDAD_ESTADO,
        NOW(),
        NOW()
    FROM STAGE.STG_PRODUCTOS_CRM PROD
    INNER JOIN ODS.ODS_DM_PAISES PAI ON CASE WHEN TRIM(PRODUCT_COUNTRY) <> '' THEN UPPER(REPLACE(PRODUCT_COUNTRY, 'United States', 'US')) ELSE 'DESCONOCIDO' END = PAI.DE_PAIS
    INNER JOIN ODS.ODS_DM_CIUDADES_ESTADOS CIU ON CASE WHEN LENGTH(TRIM(PRODUCT_CITY)) <> 0 THEN UPPER(PRODUCT_CITY) ELSE 'DESCONOCIDO' END = CIU.DE_CIUDAD AND
        CASE WHEN LENGTH(TRIM(PRODUCT_STATE)) <> 0 THEN UPPER(PRODUCT_STATE) ELSE 'DESCONOCIDO' END = CIU.DE_ESTADO
    WHERE NOT EXISTS
    (SELECT 1
    FROM ODS.ODS_HC_DIRECCIONES DIR
    WHERE CASE WHEN TRIM(PRODUCT_ADDRESS) <> '' THEN UPPER(TRIM(PRODUCT_ADDRESS)) ELSE 'DESCONOCIDO' END = DIR.DE_DIRECCION AND
    CASE WHEN LENGTH(TRIM(PRODUCT_POSTAL_CODE)) <> 0 THEN TRIM(PRODUCT_POSTAL_CODE) ELSE 99999 END = DIR.DE_CP AND
    DIR.ID_CIUDAD_ESTADO = CIU.ID_CIUDAD_ESTADO);
COMMIT;
ANALYZE TABLE ODS_HC_DIRECCIONES;
```

Poblamos el modelo de SERVICIOS (IV)

```
USE ODS;

DROP TABLE IF EXISTS TMP_DIRECCIONES_SERVICIOS;
CREATE TABLE TMP_DIRECCIONES_SERVICIOS AS
    SELECT DIR.ID_DIRECCION,
        DIR.DE_DIRECCION,
        DIR.DE_CP,
        CIU.DE_CIUDAD,
        CIU.DE_ESTADO,
        PAI.DE_PAIS
    FROM ODS.ODS_HC_DIRECCIONES DIR
    INNER JOIN ODS.ODS_DM_CIUDADES_ESTADOS CIU ON DIR.ID_CIUDAD_ESTADO = CIU.ID_CIUDAD_ESTADO
    INNER JOIN ODS.ODS_DM_PAISES PAI ON CIU.ID_PAIS = PAI.ID_PAIS;
ANALYZE TABLE TMP_DIRECCIONES_SERVICIOS;

CREATE TABLE TMP_DIRECCIONES_SERVICIOS2 AS
    SELECT PRODUCTOS.PRODUCTO_ID ID_PRODUCTO,
        DIR.ID_DIRECCION
    FROM STAGE.STG_PRODUCTOS_CRM PRODUCTOS
    INNER JOIN ODS.TMP_DIRECCIONES_SERVICIOS DIR ON
        CASE WHEN LENGTH(TRIM(PRODUCTOS.PRODUCT_ADDRESS)) <> 0 THEN UPPER(TRIM(PRODUCTOS.PRODUCT_ADDRESS)) ELSE 'DESCONOCIDO' END = DIR.DE_DIRECCION AND
        CASE WHEN LENGTH(TRIM(PRODUCTOS.PRODUCT_POSTAL_CODE)) <> 0 THEN UPPER(TRIM(PRODUCTOS.PRODUCT_POSTAL_CODE)) ELSE 99999 END = DIR.DE_CP AND
        CASE WHEN LENGTH(TRIM(PRODUCTOS.PRODUCT_CITY)) <> 0 THEN UPPER(TRIM(PRODUCTOS.PRODUCT_CITY)) ELSE 'DESCONOCIDO' END = DIR.DE_CIUDAD AND
        CASE WHEN LENGTH(TRIM(PRODUCTOS.PRODUCT_STATE)) <> 0 THEN UPPER(TRIM(PRODUCTOS.PRODUCT_STATE)) ELSE 'DESCONOCIDO' END = DIR.DE_ESTADO AND
        CASE WHEN LENGTH(TRIM(PRODUCTOS.PRODUCT_COUNTRY)) <> 0 THEN REPLACE(PRODUCTOS.PRODUCT_COUNTRY,'United States', 'US') ELSE 'DESCONOCIDO' END = DIR.DE_PAIS;
ANALYZE TABLE TMP_DIRECCIONES_SERVICIOS2;
```

Poblamos el modelo de SERVICIOS (V)

```
USE ODS;

INSERT INTO ODS.ODS_HC_CLIENTES
SELECT DISTINCT CUSTOMER_ID, 'DESCONOCIDO', 'DESCONOCIDO', '99-999-9999', 99, 999999,
    9999999999, 'DESCONOCIDO', STR_TO_DATE('31/12/9999', '%d/%m/%Y'), 999, 999, NOW(), NOW()
FROM STAGE.STG_PRODUCTOS_CRM PROD
WHERE CUSTOMER_ID NOT IN
(SELECT ID_CLIENTE
FROM ODS.ODS_HC_CLIENTES);
COMMIT;

INSERT INTO ODS_HC_SERVICIOS
SELECT
    PRODUCTO_ID AS ID_SERVICIO,
    CASE WHEN LENGTH(TRIM(CUSTOMER_ID)) <> 0 THEN TRIM(UPPER(CUSTOMER_ID)) ELSE 'DESCONOCIDO' END ID_CLIENTE,
    PRODUCTOS.ID_PRODUCTO,
    CASE WHEN LENGTH(TRIM(ACCESS_POINT)) <> 0 THEN TRIM(UPPER(ACCESS_POINT)) ELSE 'DESCONOCIDO' END PUNTO_ACCESO,
    CANALES.ID_CANAL,
    CASE WHEN LENGTH(TRIM(AGENT_CODE)) <> 0 THEN TRIM(UPPER(AGENT_CODE)) ELSE 99999 END ID_AGENTE,
    DIR.ID_DIRECCION,
    CASE WHEN LENGTH(TRIM(START_DATE)) <> 0 THEN STR_TO_DATE(START_DATE, '%d/%m/%Y') ELSE STR_TO_DATE('31/12/9999', '%d/%m/%Y') END FC_INICIO,
    CASE WHEN LENGTH(TRIM(INSTALL_DATE)) <> 0 THEN DATE_FORMAT(REPLACE(INSTALL_DATE, 'UTC', ''), '%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999', '%d/%m/%Y') END FC_INSTALACION,
    CASE WHEN LENGTH(TRIM(END_DATE)) <> 0 THEN DATE_FORMAT(REPLACE(END_DATE, 'UTC', ''), '%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999', '%d/%m/%Y') END FC_FIN,
    NOW(),
    NOW()
FROM STAGE.STG_PRODUCTOS_CRM PROD
    INNER JOIN ODS.ODS_HC_CLIENTES CLIENTES ON CASE WHEN LENGTH(TRIM(PROD.CUSTOMER_ID)) <> 0 THEN UPPER(TRIM(PROD.CUSTOMER_ID)) ELSE 'DESCONOCIDO' END = CLIENTES.ID_CLIENTE
    INNER JOIN ODS.ODS_DM_PRODUCTOS PRODUCTOS ON CASE WHEN LENGTH(TRIM(PROD.PRODUCT_NAME)) <> 0 THEN UPPER(TRIM(PROD.PRODUCT_NAME)) ELSE 'DESCONOCIDO' END = PRODUCTOS.DE_PRODUCTO
    INNER JOIN ODS.ODS_DM_CANALES CANALES ON CASE WHEN LENGTH(TRIM(PROD.CHANNEL)) <> 0 THEN UPPER(TRIM(PROD.CHANNEL)) ELSE 'DESCONOCIDO' END = CANALES.DE_CANAL
    INNER JOIN ODS.TMP_DIRECCIONES_SERVICIOS2 DIR ON PROD.PRODUCTO_ID = DIR.ID_PRODUCTO;
COMMIT;
ANALYZE TABLE ODS_HC_CLIENTES;

DROP TABLE IF EXISTS TMP_DIRECCIONES_CLIENTES;
DROP TABLE IF EXISTS TMP_DIRECCIONES_CLIENTES2;
COMMIT;
```

Estudio de STG_FACTURAS_FCT (I)

```sql
USE STAGE;

SELECT COUNT(*) TOTAL_REGISTROS,
SUM(CASE WHEN LENGTH(TRIM(BILL_REF_NO)) <> 0 THEN 1 ELSE 0 END) TOTAL_BILL_REF_NO,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(BILL_REF_NO)) <> 0 THEN BILL_REF_NO ELSE 0 END) TOTAL_DISTINTOS_BILL_REF_NO,
SUM(CASE WHEN LENGTH(TRIM(CUSTOMER_ID)) <> 0 THEN 1 ELSE 0 END) TOTAL_CUSTOMER_ID,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(CUSTOMER_ID)) <> 0 THEN CUSTOMER_ID ELSE 0 END) TOTAL_DISTINTOS_CUSTOMER_ID,
SUM(CASE WHEN LENGTH(TRIM(START_DATE)) <> 0 THEN 1 ELSE 0 END) TOTAL_START_DATE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(START_DATE)) <> 0 THEN START_DATE ELSE 0 END) TOTAL_DISNTINTOS_START_DATE,
SUM(CASE WHEN LENGTH(TRIM(END_DATE)) <> 0 THEN 1 ELSE 0 END) TOTAL_END_DATE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(END_DATE)) <> 0 THEN END_DATE ELSE 0 END) TOTAL_DISNTINTOS_END_DATE,
SUM(CASE WHEN LENGTH(TRIM(STATEMENT_DATE)) <> 0 THEN 1 ELSE 0 END) TOTAL_STATEMENT_DATE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(STATEMENT_DATE)) <> 0 THEN STATEMENT_DATE ELSE 0 END) TOTAL_DISNTINTOS_STATEMENT_DATE,
SUM(CASE WHEN LENGTH(TRIM(PAYMENT_DATE)) <> 0 THEN 1 ELSE 0 END) TOTAL_PAYMENT_DATE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(PAYMENT_DATE)) <> 0 THEN PAYMENT_DATE ELSE 0 END) TOTAL_DISNTINTOS_PAYMENT_DATE,
SUM(CASE WHEN LENGTH(TRIM(BILL_CYCLE)) <> 0 THEN 1 ELSE 0 END) TOTAL_BILL_CYCLE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(BILL_CYCLE)) <> 0 THEN BILL_CYCLE ELSE 0 END) TOTAL_DISNTINTOS_BILL_CYCLE,
SUM(CASE WHEN LENGTH(TRIM(AMOUNT)) <> 0 THEN 1 ELSE 0 END) TOTAL_AMOUNT,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(AMOUNT)) <> 0 THEN AMOUNT ELSE 0 END) TOTAL_DISNTINTOS_AMOUNT,
SUM(CASE WHEN LENGTH(TRIM(BILL_METHOD)) <> 0 THEN 1 ELSE 0 END) TOTAL_BILL_METHOD,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(BILL_METHOD)) <> 0 THEN BILL_METHOD ELSE 0 END) TOTAL_DISNTINTOS_BILL_METHOD
FROM STAGE.STG_FACTURAS_FCT;
```

Estudio de STG_FACTURAS_FCT (II)

TOTAL_REGUSTROS:	420000
TOTAL_BILL_REF_NO:	420000
TOTAL_DISTINTOS_BILL_REF_NO:	420000
TOTAL_CUSTOMER_ID:	420000
TOTAL_DISTINTOS_CUSTOMER_ID:	20000
TOTAL_START_DATE:	420000
TOTAL_DISNTINTOS_START_DATE:	40
TOTAL_END_DATE:	420000
TOTAL_DISNTINTOS_END_DATE:	20
TOTAL_STATEMENT_DATE:	420000
TOTAL_DISNTINTOS_STATEMENT_DATE:	40
TOTAL_PAYMENT_DATE:	420000
TOTAL_DISNTINTOS_PAYMENT_DATE:	400
TOTAL_BILL_CYCLE:	420000
TOTAL_DISNTINTOS_BILL_CYCLE:	2
TOTAL_AMOUNT:	420000
TOTAL_DISNTINTOS_AMOUNT:	5604
TOTAL_BILL_METHOD:	420000
TOTAL_DISNTINTOS_BILL_METHOD:	3

Estudio de STG_FACTURAS_FCT (III)

Conclusiones:

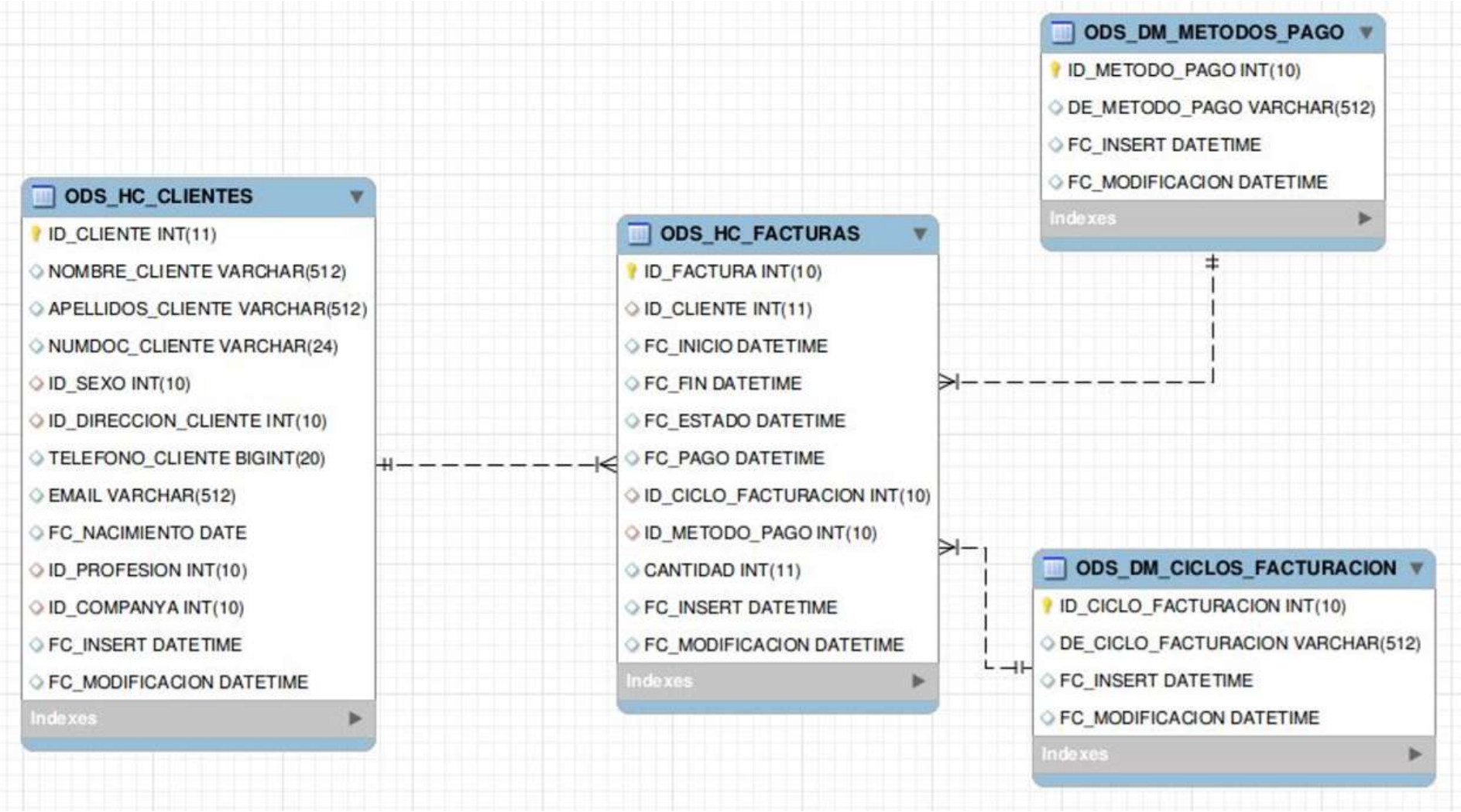
- Dado que el campo BILL_REF_NO no se repite en ningún registro, y además, no presenta valores nulos, será el que usaremos como Clave Principal de la tabla de ODS_HC_FACTURAS

- El campo CUSTOMER_ID será la Clave Ajena respecto la tabla ODS_HC_CLIENTES
  - Hay 2442 registros cuyos clientes no se encuentran en ODS_HC_CLIENTE

- Dado que sólo hay 2 ciclos de facturación diferentes, deberíamos crear tabla dimensional para almacenarlos

- Del mismo modo, dado que sólo hay 3 métodos de pago diferentes, deberíamos crear tabla dimensional para almacenarlos

Creamos modelo de FACTURAS



Creamos las tablas de FACTURAS

```
USE ODS;

DROP TABLE IF EXISTS ODS_HC_FACTURAS;
CREATE TABLE IF NOT EXISTS ODS_HC_FACTURAS (
    ID_FACTURA INT(10) UNSIGNED NOT NULL PRIMARY KEY,
    ID_CLIENTE INT(11),
    FC_FECHA_INICIO DATETIME,
    FC_FECHA_FIN DATETIME,
    FC_ESTADO DATETIME,
    FC_PAGO DATETIME,
    ID_CICLO_FACTURACION INT(10),
    ID_METODO_PAGO INT(10),
    CANTIDAD INT(11),
    FC_INSERT DATETIME,
    FC_MODIFICACION DATETIME);

DROP TABLE IF EXISTS ODS_DM_METODOS_PAGO;
CREATE TABLE IF NOT EXISTS ODS_DM_METODOS_PAGO (
    ID_METODO_PAGO INT(10) UNSIGNED AUTO_INCREMENT NOT NULL PRIMARY KEY,
    DE_METODO_PAGO VARCHAR(512),
    FC_INSERT DATETIME,
    FC_MODIFICACION DATETIME);

DROP TABLE IF EXISTS ODS_DM_CICLOS_FACTURACION;
CREATE TABLE IF NOT EXISTS ODS_DM_CICLOS_FACTURACION (
    ID_CICLO_FACTURACION INT(10) UNSIGNED AUTO_INCREMENT NOT NULL PRIMARY KEY,
    DE_CICLO_FACTURACION VARCHAR(512),
    FC_INSERT DATETIME,
    FC_MODIFICACION DATETIME);
```

Creamos las FK del modelo de FACTURAS

```
USE ODS;

ALTER TABLE ODS_HC_FACTURAS MODIFY COLUMN ID_CLIENTE INT(11);
ALTER TABLE ODS_HC_FACTURAS ADD INDEX fk_fact_cli_idx (ID_CLIENTE ASC);
ALTER TABLE ODS_HC_FACTURAS ADD CONSTRAINT fk_fact_cli FOREIGN KEY (ID_CLIENTE)
REFERENCES ODS_HC_CLIENTES (ID_CLIENTE);

ALTER TABLE ODS_HC_FACTURAS MODIFY COLUMN ID_METODO_PAGO INT(10) UNSIGNED;
ALTER TABLE ODS_HC_FACTURAS ADD INDEX fk_fact_meth_idx (ID_METODO_PAGO ASC);
ALTER TABLE ODS_HC_FACTURAS ADD CONSTRAINT fk_fact_meth FOREIGN KEY (ID_METODO_PAGO)
REFERENCES ODS_DM_METODOS_PAGO (ID_METODO_PAGO);

ALTER TABLE ODS_HC_FACTURAS MODIFY COLUMN ID_CICLO_FACTURACION INT(10) UNSIGNED;
ALTER TABLE ODS_HC_FACTURAS ADD INDEX fk_fact_cycl_idx (ID_CICLO_FACTURACION ASC);
ALTER TABLE ODS_HC_FACTURAS ADD CONSTRAINT fk_fact_cycl FOREIGN KEY (ID_CICLO_FACTURACION)
REFERENCES ODS_DM_CICLOS_FACTURACION (ID_CICLO_FACTURACION);
```

Poblamos el modelo de FACTURAS (I)

```
USE ODS;

INSERT INTO ODS_DM_METODOS_PAGO (DE_METODO_PAGO, FC_INSERT, FC_MODIFICACION)
    SELECT DISTINCT UPPER(TRIM(BILL_METHOD)), NOW(), NOW()
    FROM STAGE.STG_FACTURAS_FCT
    WHERE LENGTH(TRIM(BILL_METHOD)) <> 0;
COMMIT;
INSERT INTO ODS_DM_METODOS_PAGO VALUES (99, 'DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS_DM_METODOS_PAGO VALUES (98, 'NO APLICA', NOW(), NOW());
COMMIT;
ANALYZE TABLE ODS_DM_METODOS_PAGO;

INSERT INTO ODS_DM_CICLOS_FACTURACION (DE_CICLO_FACTURACION, FC_INSERT, FC_MODIFICACION)
    SELECT DISTINCT UPPER(TRIM(BILL_CYCLE)), NOW(), NOW()
    FROM STAGE.STG_FACTURAS_FCT
    WHERE LENGTH(TRIM(BILL_CYCLE)) <> 0;
COMMIT;
INSERT INTO ODS_DM_CICLOS_FACTURACION VALUES (99, 'DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS_DM_CICLOS_FACTURACION VALUES (98, 'NO APLICA', NOW(), NOW());
COMMIT;
ANALYZE TABLE ODS_DM_CICLOS_FACTURACION;
USE ODS;
```

Poblamos el modelo de FACTURAS (II)

```
USE ODS;

INSERT INTO ODS.ODS_HC_CLIENTES
SELECT DISTINCT CUSTOMER_ID, 'DESCONOCIDO', 'DESCONOCIDO', '99-999-9999', 99, 999999,
    9999999999, 'DESCONOCIDO',  STR_TO_DATE('31/12/9999', '%d/%m/%Y'), 999,  999, NOW(), NOW()
FROM STAGE.STG_FACTURAS_FCT FCT
WHERE CUSTOMER_ID NOT IN
(SELECT ID_CLIENTE
FROM ODS.ODS_HC_CLIENTES);
COMMIT;

INSERT INTO ODS_HC_FACTURAS
SELECT
    BILL_REF_NO AS ID_FACTURA,
    CLI.ID_CLIENTE,
    CASE WHEN LENGTH(TRIM(START_DATE)) <> 0 THEN DATE_FORMAT(START_DATE, '%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999', '%d/%m/%Y') END FC_INICIO,
    CASE WHEN LENGTH(TRIM(END_DATE)) <> 0 THEN DATE_FORMAT(END_DATE, '%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999', '%d/%m/%Y') END FC_FIN,
    CASE WHEN LENGTH(TRIM(STATEMENT_DATE)) <> 0 THEN DATE_FORMAT(STATEMENT_DATE, '%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999', '%d/%m/%Y') END FC_ESTADO,
    CASE WHEN LENGTH(TRIM(PAYMENT_DATE)) <> 0 THEN DATE_FORMAT(PAYMENT_DATE, '%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999', '%d/%m/%Y') END FC_PAGO,
    CYCL.ID_CICLO_FACTURACION,
    METH.ID_METODO_PAGO,
    CASE WHEN LENGTH(TRIM(AMOUNT)) <> 0 THEN TRIM(UPPER(AMOUNT)) ELSE 0 END CANTIDAD,
    NOW(),
    NOW()
FROM STAGE.STG_FACTURAS_FCT
    INNER JOIN ODS.ODS_HC_CLIENTES CLI ON CASE WHEN LENGTH(TRIM(CUSTOMER_ID)) <> 0 THEN UPPER(TRIM(CUSTOMER_ID)) ELSE 'DESCONOCIDO' END = CLI.ID_CLIENTE
    INNER JOIN ODS.ODS_DM_CICLOS_FACTURACION CYCL ON CASE WHEN LENGTH(TRIM(BILL_CYCLE)) <> 0 THEN UPPER(TRIM(BILL_CYCLE)) ELSE 'DESCONOCIDO' END = CYCL.DE_CICLO_FACTURACION
    INNER JOIN ODS.ODS_DM_METODOS_PAGO METH ON CASE WHEN LENGTH(TRIM(BILL_METHOD)) <> 0 THEN UPPER(TRIM(BILL_METHOD)) ELSE 'DESCONOCIDO' END = METH.DE_METODO_PAGO;
COMMIT;
ANALYZE TABLE ODS_HC_FACTURAS;
```

Estudio de STG_CONTACTOS_IVR (I)

```sql
USE STAGE;

SELECT COUNT(*) TOTAL_REGISTROS,
SUM(CASE WHEN LENGTH(TRIM(ID)) <> 0 THEN 1 ELSE 0 END) TOTAL_ID,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(ID)) <> 0 THEN ID ELSE 0 END) TOTAL_DISTINTOS_ID,
SUM(CASE WHEN LENGTH(TRIM(PHONE_NUMBER)) <> 0 THEN 1 ELSE 0 END) TOTAL_PHONE_NUMBER,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(PHONE_NUMBER)) <> 0 THEN PHONE_NUMBER ELSE 0 END) TOTAL_DISTINTOS_PHONE_NUMBER,
SUM(CASE WHEN LENGTH(TRIM(START_DATETIME)) <> 0 THEN 1 ELSE 0 END) TOTAL_START_DATETIME,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(START_DATETIME)) <> 0 THEN START_DATETIME ELSE 0 END) TOTAL_DISTINTOS_START_DATETIME,
SUM(CASE WHEN LENGTH(TRIM(END_DATETIME)) <> 0 THEN 1 ELSE 0 END) TOTAL_END_DATETIME,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(END_DATETIME)) <> 0 THEN END_DATETIME ELSE 0 END) TOTAL_DISTINTOS_END_DATETIME,
SUM(CASE WHEN LENGTH(TRIM(SERVICE)) <> 0 THEN 1 ELSE 0 END) TOTAL_SERVICE,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(SERVICE)) <> 0 THEN SERVICE ELSE 0 END) TOTAL_DISTINTOS_SERVICE,
SUM(CASE WHEN LENGTH(TRIM(FLG_TRANSFER)) <> 0 THEN 1 ELSE 0 END) TOTAL_FLG_TRANSFER,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(FLG_TRANSFER)) <> 0 THEN FLG_TRANSFER ELSE 0 END) TOTAL_DISTINTOS_FLG_TRANSFER,
SUM(CASE WHEN LENGTH(TRIM(AGENT)) <> 0 THEN 1 ELSE 0 END) TOTAL_AGENT,
COUNT(DISTINCT CASE WHEN LENGTH(TRIM(AGENT)) <> 0 THEN AGENT ELSE 0 END) TOTAL_DISTINTOS_AGENT
FROM STAGE.STG_CONTACTOS_IVR CON;
```

Estudio de STG_CONTACTOS_IVR (II)

TOTAL_REGISTROS:	202717
TOTAL_ID:	202717
TOTAL_DISTINTOS_ID:	150000
TOTAL_PHONE_NUMBER:	185018
TOTAL_DISTINTOS_PHONE_NUMBER:	18226
TOTAL_START_DATETIME:	202717
TOTAL_DISTINTOS_START_DATETIME:	201098
TOTAL_END_DATETIME:	186535
TOTAL_DISTINTOS_END_DATETIME:	183678
TOTAL_SERVICE:	202502
TOTAL_DISTINTOS_SERVICE:	7
TOTAL_FLG_TRANSFER:	202717
TOTAL_DISTINTOS_FLG_TRANSFER:	2
TOTAL_AGENT:	194739
TOTAL_DISTINTOS_AGENT:	594

Estudio de STG_CONTACTOS_IVR (III)

Conclusiones:

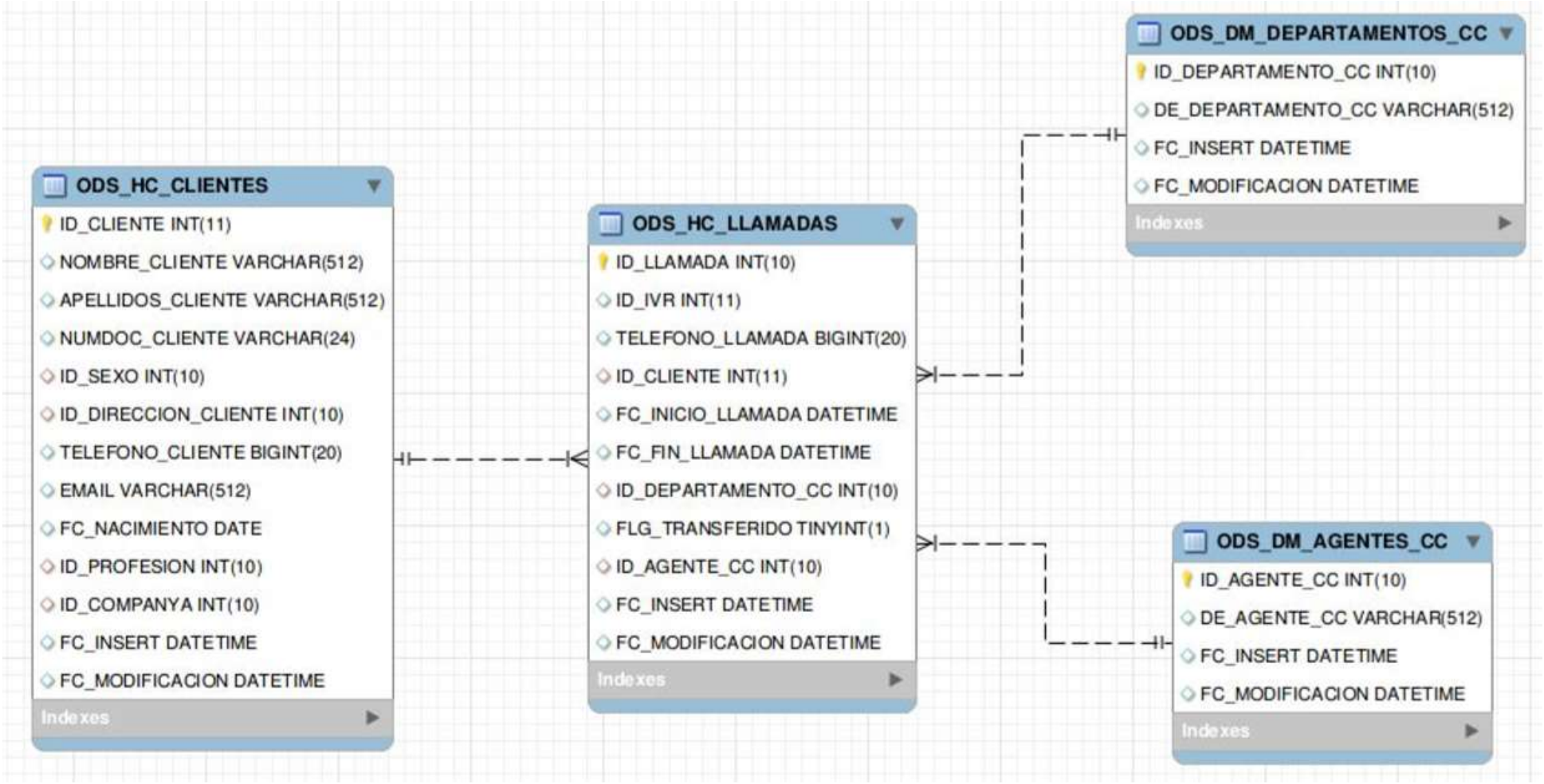
- Dado que el campo ID se repite en varios registros, y además, no podemos usarlo como Clave Principal de la tabla de LLAMADAS
  - Crearemos un campo, ID_LLAMADAS que usaremos de PK
  - Almacenaremos el valor del ID en el campo ID_IVR para mantener trazabilidad de este dato

- El campo PHONE_NUMBER debería hacer referencia a teléfonos de clientes de la tabla ODS_HC_CLIENTES, pero no coincide ningún número, así que, añadiremos el ID_CLIENTE como clave ajena, y lo iniciaremos como cliente desconocido (999999)

- Dado que sólo hay 6 DEPARTAMENTOS diferentes, almacenados en el campo SERVICE, deberíamos crear tabla dimensional para almacenarlos

- También crearemos una tabla dimensional para los AGENT_CODE
  - Hay 7978 registros con AGENT en blanco

Creamos modelo de LLAMADAS



Creamos las tablas de LLAMADAS

```
USE ODS;

DROP TABLE IF EXISTS ODS_HC_LLAMADAS;
CREATE TABLE IF NOT EXISTS ODS_HC_LLAMADAS (
    ID_LLAMADA INT(10) AUTO_INCREMENT NOT NULL PRIMARY KEY,
    ID_IVR INT(10),
    TELEFONO_LLAMADA BIGINT(20),
    ID_CLIENTE INT(11),
    FC_INICIO_LLAMADA DATETIME,
    FC_FIN_LLAMADA DATETIME,
    ID_DEPARTAMENTO_CC INT(10) UNSIGNED,
    FLG_TRANSFERIDO TINYINT(1),
    ID_AGENTE_CC INT(10) UNSIGNED,
    FC_INSERT DATETIME,
    FC_MODIFICACION DATETIME);

DROP TABLE IF EXISTS ODS_DM_DEPARTAMENTOS_CC;
CREATE TABLE IF NOT EXISTS ODS_DM_DEPARTAMENTOS_CC (
    ID_DEPARTAMENTO_CC INT(10) AUTO_INCREMENT NOT NULL PRIMARY KEY,
    DE_DEPARTAMENTO_CC VARCHAR(512),
    FC_INSERT DATETIME,
    FC_MODIFICACION DATETIME);

DROP TABLE IF EXISTS ODS_DM_AGENTES_CC;
CREATE TABLE IF NOT EXISTS ODS_DM_AGENTES_CC (
    ID_AGENTE_CC INT(10) AUTO_INCREMENT NOT NULL PRIMARY KEY,
    DE_AGENTE_CC VARCHAR(512),
    FC_INSERT DATETIME,
    FC_MODIFICACION DATETIME);
```

Creamos las FK del modelo de LLAMADAS

```
USE ODS;

ALTER TABLE ODS_HC_LLAMADAS MODIFY COLUMN ID_CLIENTE INT(11);
ALTER TABLE ODS_HC_LLAMADAS ADD INDEX fk_llam_cli_idx (ID_CLIENTE ASC);
ALTER TABLE ODS_HC_LLAMADAS ADD CONSTRAINT fk_llam_cli FOREIGN KEY (ID_CLIENTE)
REFERENCES ODS_HC_CLIENTES (ID_CLIENTE);

ALTER TABLE ODS_HC_LLAMADAS MODIFY COLUMN ID_DEPARTAMENTO_CC INT(10);
ALTER TABLE ODS_HC_LLAMADAS ADD INDEX fk_llam_dep_idx (ID_DEPARTAMENTO_CC ASC);
ALTER TABLE ODS_HC_LLAMADAS ADD CONSTRAINT fk_llam_dep FOREIGN KEY (ID_DEPARTAMENTO_CC)
REFERENCES ODS_DM_DEPARTAMENTOS_CC (ID_DEPARTAMENTO_CC);

ALTER TABLE ODS_HC_LLAMADAS MODIFY COLUMN ID_AGENTE_CC INT(10);
ALTER TABLE ODS_HC_LLAMADAS ADD INDEX fk_llam_age_idx (ID_AGENTE_CC ASC);
ALTER TABLE ODS_HC_LLAMADAS ADD CONSTRAINT fk_llam_age FOREIGN KEY (ID_AGENTE_CC)
REFERENCES ODS_DM_AGENTES_CC (ID_AGENTE_CC);
```

Poblamos el modelo de LLAMADAS (I)

```
USE ODS;

INSERT INTO ODS_DM_DEPARTAMENTOS_CC (DE_DEPARTAMENTO_CC, FC_INSERT, FC_MODIFICACION)
    SELECT DISTINCT UPPER(TRIM(SERVICE)), NOW(), NOW()
    FROM STAGE.STG_CONTACTOS_IVR
    WHERE LENGTH(TRIM(SERVICE)) <> 0;
COMMIT;
INSERT INTO ODS_DM_DEPARTAMENTOS_CC VALUES (99, 'DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS_DM_DEPARTAMENTOS_CC VALUES (98, 'NO APLICA', NOW(), NOW());
COMMIT;
ANALYZE TABLE ODS_DM_DEPARTAMENTOS_CC;

INSERT INTO ODS_DM_AGENTES_CC (DE_AGENTE_CC, FC_INSERT, FC_MODIFICACION)
    SELECT DISTINCT UPPER(TRIM(AGENT)), NOW(), NOW()
    FROM STAGE.STG_CONTACTOS_IVR
    WHERE LENGTH(TRIM(AGENT)) <> 0;
COMMIT;
INSERT INTO ODS_DM_AGENTES_CC VALUES (999, 'DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS_DM_AGENTES_CC VALUES (998, 'NO APLICA', NOW(), NOW());
COMMIT;
ANALYZE TABLE ODS_DM_AGENTES_CC;
```

Poblamos el modelo de LLAMADAS (II)

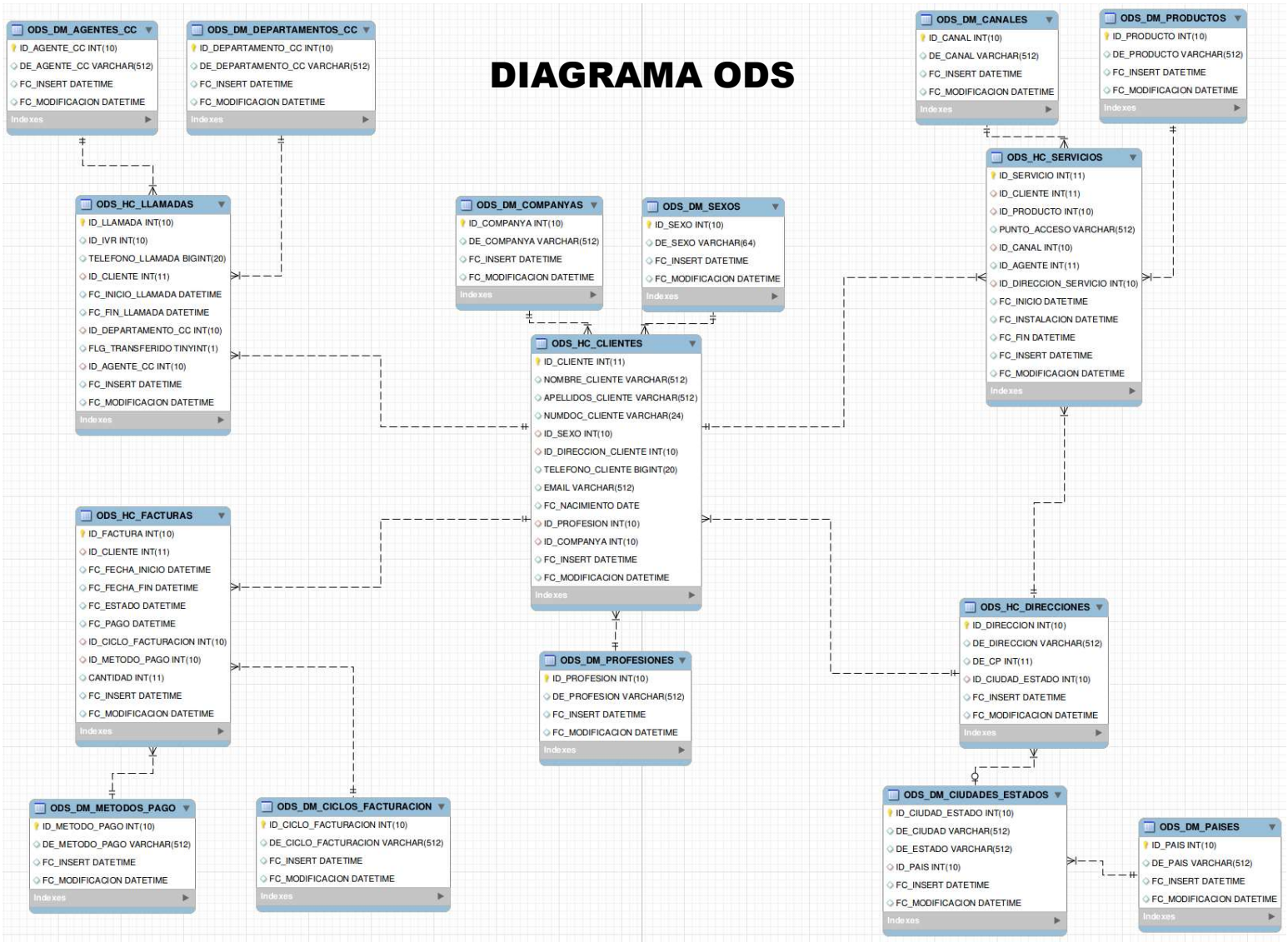
```
USE ODS;

INSERT INTO ODS.ODS_HC_CLIENTES
VALUES (999999, 'DESCONOCIDO', 'DESCONOCIDO', '99-999-9999', 99, 999999, 9999999999,
    'DESCONOCIDO', STR_TO_DATE('31/12/9999', '%d/%m/%Y'), 999,  999,  NOW(),  NOW());
INSERT INTO ODS.ODS_HC_CLIENTES
VALUES (999998, 'NO APLICA', 'NO APLICA', '99-999-9998', 98, 999998, 9999999998,
    'NO APLICA', STR_TO_DATE('31/12/9998', '%d/%m/%Y'), 998,  998,  NOW(),  NOW());
COMMIT;

INSERT INTO ODS_HC_LLAMADAS (ID_IVR, TELEFONO_LLAMADA,ID_CLIENTE,FC_INICIO_LLAMADA,FC_FIN_LLAMADA,ID_DEPARTAMENTO_CC,FLG_TRANSFERIDO,ID_AGENTE_CC,FC_INSERT,FC_MODIFICACION)
SELECT
    ID AS ID_IVR,
    CASE WHEN LENGTH(TRIM(PHONE_NUMBER)) <> 0 THEN TRIM(UPPER(PHONE_NUMBER)) ELSE 0 END TELEFONO_LLAMADA,
    999999,
    CASE WHEN LENGTH(TRIM(START_DATETIME)) <> 0 THEN DATE_FORMAT(START_DATETIME, '%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999', '%d/%m/%Y') END FC_INICIO_LLAMADA,
    CASE WHEN LENGTH(TRIM(END_DATETIME)) <> 0 THEN DATE_FORMAT(END_DATETIME, '%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999', '%d/%m/%Y') END FC_FIN_LLAMADA,
    DEP.ID_DEPARTAMENTO_CC,
    CASE WHEN FLG_TRANSFER = 'True' THEN 1 ELSE 0 END FLG_TRANSFERIDO,
    AGE.ID_AGENTE_CC,
    NOW(),
    NOW()
FROM STAGE.STG_CONTACTOS_IVR
INNER JOIN ODS.ODS_DM_DEPARTAMENTOS_CC DEP ON CASE WHEN LENGTH(TRIM(SERVICE)) <> 0 THEN UPPER(TRIM(SERVICE)) ELSE 'DESCONOCIDO' END = DEP.DE_DEPARTAMENTO_CC
INNER JOIN ODS.ODS_DM_AGENTES_CC AGE ON CASE WHEN LENGTH(TRIM(AGENT)) <> 0 THEN UPPER(TRIM(AGENT)) ELSE 'DESCONOCIDO' END = AGE.DE_AGENTE_CC;
COMMIT;
ANALYZE TABLE ODS_HC_LLAMADAS;
```

SEGUNDA PARTE

DIAGRAMA ODS



¿Por qué en el modelo de DIRECCIONES dejo en la misma tabla las CIUDADES y los ESTADOS y no los separo en dos tablas distintas para ser más estricta con la jerarquía: PAIS → ESTADOS → CIUDADES → DIRECCIONES?

- Debido a la posibilidad de encontrarnos con Ciudades del mismo nombre que pertenezcan a distintos Estados
- Por ejemplo, en la tabla STG_CLIENTES_CRM hay una ciudad, Glendale, que existe tanto en el estado de California como en el de Arizona

TERCERA PARTE

DATA QUALITY

- En la tabla STG_PRODUCTOS_CRM, había 3 CUSTOMER_ID que no existían en la tabla de STG_CLIENTES_CRM
  En la tabla STG_FACTURAS_FCT había 2442 CUSTOMER_ID no localizables en STG_CLIENTES_CRM
  En el caso de la tabla STG_CONTACTOS_IVR, no tenemos forma de cruzar sus datos con STG_CLIENTES_CRM ya que ni siquiera los números de teléfono introducidos pertenecen a los clientes existentes
  Se debería controlar que al introducir un ID de cliente, éste existiera, para evitar este tipo de situaciones

- En la tabla STG_PRODUCTOS_CRM hemos tenido que habilitar un tratamiento para las fechas, ya que su formato no era compatible con la tabla destino, y nos impedía insertarlas
  Lo mismo ocurre con la tabla STG_FACTURAS_FCT, que usa un formato de fechas distinto al usado en STG_CLIENTES_CRM
  Debería haber un formato de fechas prestablecido para todas las tablas

- También en STG_PRODUCTOS_CRM hemos determinado que en realidad no se hace referencia a productos como tal, sino a servicios instalados por la empresa
  En la tabla STG_CONTACTOS_IVR, el campo SERVICE en realidad se refiere al Departamento
  El nombre de los campos y tablas deberían ser revisados según los valores almacenados en ellos

- Tanto en la tabla STG_PRODUCTOS_CRM como en la tabla STG_CONTACTOS_IVR nos encontramos con datos que hacen referencia al agente que lleva la operación, sin que tengamos más información acerca de los mismos
  Debería incorporarse datos sobre los agentes en activo en la empresa

- En la tabla de STG_CONTACTOS_IVR nos hemos encontrado con que había valores duplicados en el campo ID, lo cual nos impedía usarlo como Primary Key

MASTER DATA

- En la tabla STG_PRODUCTOS_CRM, nos hemos encontrado con que la cadena usada para indicar el país era ‘United States’ (esto es inconsistente con el país indicado en la tabla STG_CLIENTES_CRM, que es ‘US’)
  Si hubiera una tabla maestra en la que se determinaran los códigos a usar por países, esto no ocurriría.

- En el caso de las Direcciones, también sería interesante tener tablas maestras para códigos de Estados, Ciudades, o Códigos Postales

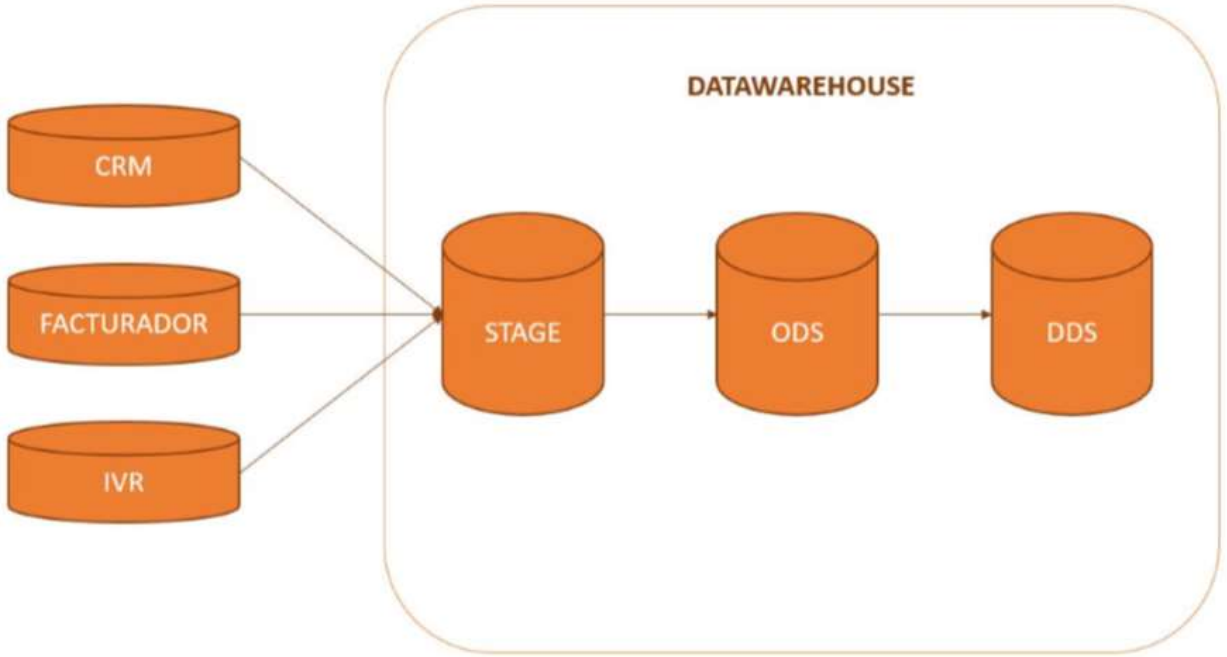
DATA MODELING AND DESIGN

- Lo dejaría tal cual está

- Si acaso, añadiría tablas maestras para datos dimensionales que pueden ser compartidas por las diversas áreas para estandarizar sus diferentes valores posibles, y poder relacionarse correctamente entre ellas

CUARTA PARTE

Después de todo lo visto nuestro ecosistema quedaría así:



QUINTA PARTE

Escribe tus propias reglas o mandamientos de un DataWarehouse:

- Replicar las bases de datos de negocio en STAGE, en momentos de baja carga operacional, para no interferir en su correcto funcionamiento

- Considero importante el disponer de una serie de tablas maestras para los datos relativos a direcciones, países, o cualquier dato dimensional susceptible de ser utilizado por diferentes departamentos/modelos con el fin de que sean datos cuyos códigos sean comunes, evitando posibles errores y facilitando la correcta relación entre las diferentes tablas

- Desde STAGE, generamos ODS, creando las tablas dimensionales y de hechos correspondientes
  - Procuraremos que el formato de los campos coincida para poder relacionarlos correctamente
  - Evitaremos el uso de Nulos en campos clave
  - Relacionamos tablas con claves ajenas, procurando usar INNER JOIN a la hora de poblar, ya que es más eficiente que usar OUTER JOIN
  - Los distintos modelos creados (en nuestro caso: Clientes, Servicios, Facturas y Llamadas), deben estar relacionados entre sí para evitar la existencia de islas de datos

SEXTA PARTE

¿Nivel de SQL antes y después?

	ANTES	DESPUÉS
Estoy más perdido que un muelle en las escaleras de Hogwarts		
Dejad que las queries se acerquen a mí		
Todo lo que quiso saber sobre SQL y no se atrevió a preguntar	✓	✓
TRUNCATE TABLE "sql_problems"		