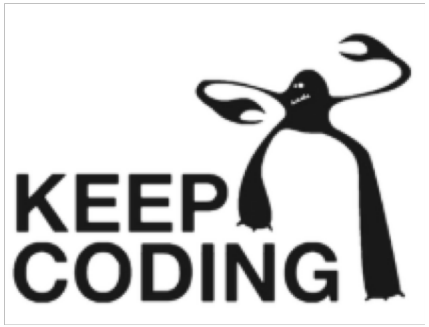
Práctica Taller DataWarehouse

Ramón Lerena Villarroel



FACTURAS

 # 1) Estudio de las tablas

 # Tabla STG_FACTURAS_FCT

```
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_DISTINTOS_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_DISTINTOS_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_DISTINTOS_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_DISTINTOS_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_DISTINTOS_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_DISTINTOS_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_DISTINTOS_BILL_METHOD
FROM STAGE.STG_FACTURAS_FCT;
```

TOTAL_REGISTROS:	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_DISTINTOS_START_DATE:	40
TOTAL_END_DATE:	420000
TOTAL_DISTINTOS_END_DATE:	20
TOTAL_STATEMENT_DATE:	420000
TOTAL_DISTINTOS_STATEMENT_DATE:	40
TOTAL_PAYMENT_DATE:	420000
TOTAL_DISTINTOS_PAYMENT_DATE:	400
TOTAL_BILL_CYCLE:	420000
TOTAL_DISTINTOS_BILL_CYCLE:	2
TOTAL_AMOUNT:	420000
TOTAL_DISTINTOS_AMOUNT:	5604
TOTAL_BILL_METHOD:	420000
TOTAL_DISTINTOS_BILL_METHOD:	3

LLAMADAS

 # 1) Estudio de las tablas

 # Tabla STG_CONTACTOS_IVR

```
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;
```

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

DESARROLLO MODELO FACTURAS

 # 1) Creación de las tablas en ODS

 # 1.1 Creación tabla dimensión Métodos de pago

```
DROP TABLE IF EXISTS ODS.ODS_DM_METODOS_PAGO;
CREATE TABLE ODS.ODS_DM_METODOS_PAGO(ID_METODO_PAGO
INT(10) UNSIGNED AUTO_INCREMENT PRIMARY KEY
, DE_METODO_PAGO VARCHAR(512)
, FC_INSERT DATETIME
, FC_MODIFICACION DATETIME);

ANALYZE TABLE ODS.ODS_DM_METODOS_PAGO;
```

 # 1.2 Creación tabla dimensión Ciclos de facturación

```
DROP TABLE IF EXISTS ODS.ODS_DM_CICLOS_FACTURACION;
CREATE TABLE ODS.ODS_DM_CICLOS_FACTURACION
(ID_CICLO_FACTURACION INT(10) UNSIGNED AUTO_INCREMENT PRIMARY KEY
, DE_CICLO_FACTURACION VARCHAR(512)
, FC_INSERT DATETIME
, FC_MODIFICACION DATETIME);

ANALYZE TABLE ODS.ODS_DM_CICLOS_FACTURACION;
```

 # 1.3 Creación tabla hechos Facturas

```
DROP TABLE IF EXISTS ODS.ODS_HC_FACTURAS;
CREATE TABLE ODS.ODS_HC_FACTURAS
(ID_FACTURA INT(10) UNSIGNED AUTO_INCREMENT PRIMARY KEY
, ID_CLIENTE INT(11)
, FC_INICIO DATE
, FC_FIN DATE
, FC_ESTADO DATE
, FC_PAGO DATE
, ID_CICLO_FACTURACION INT(10) UNSIGNED
, ID_METODO_PAGO INT(10) UNSIGNED
, CANTIDAD INT(11)
, FC_INSERT DATETIME
, FC_MODIFICACION DATETIME);

ANALYZE TABLE ODS.ODS_HC_FACTURAS;
```

 # 2) Creación FKs

```
ALTER TABLE ODS.ODS_HC_FACTURAS ADD INDEX fk_fact_ciclo_idx (ID_CICLO_FACTURACION ASC);
ALTER TABLE ODS.ODS_HC_FACTURAS ADD CONSTRAINT fk_fac_ciclo FOREIGN KEY(ID_CICLO_FACTURACION)
  REFERENCES ODS.ODS_DM_CICLOS_FACTURACION(ID_CICLO_FACTURACION);

ALTER TABLE ODS.ODS_HC_FACTURAS ADD INDEX fk_fact_met_idx (ID_METODO_PAGO ASC);
ALTER TABLE ODS.ODS_HC_FACTURAS ADD CONSTRAINT fk_fac_met FOREIGN KEY(ID_METODO_PAGO)
  REFERENCES ODS.ODS_DM_METODOS_PAGO(ID_METODO_PAGO);

ALTER TABLE ODS.ODS_HC_FACTURAS ADD INDEX fk_fac_cli_idx (ID_CLIENTE ASC);
ALTER TABLE ODS.ODS_HC_FACTURAS ADD CONSTRAINT fk_fac_cli FOREIGN KEY(ID_CLIENTE)
    REFERENCES ODS.ODS_HC_CLIENTES(ID_CLIENTE);
```

3) Poblamos el modelo

3.1 Poblamos tabla dimensión Métodos de pago

```
INSERT INTO ODS.ODS_DM_METODOS_PAGO (DE_METODO_PAGO, FC_INSERT, FC_MODIFICACION)
SELECT DISTINCT UPPER(TRIM(BILL_METHOD)) METODO_PAGO
, NOW(), NOW()
FROM STAGE.STG_FACTURAS_FCT FACT
WHERE TRIM(BILL_METHOD)<>'';
INSERT INTO ODS.ODS_DM_METODOS_PAGO VALUES(9999, 'DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS.ODS_DM_METODOS_PAGO VALUES(9998, 'NO APLICA', NOW(), NOW());
COMMIT;

ANALYZE TABLE ODS.ODS_DM_METODOS_PAGO;
```

3.2 Poblamos tabla dimensión Ciclos de facturación

```
INSERT INTO ODS.ODS_DM_CICLOS_FACTURACION (DE_CICLO_FACTURACION, FC_INSERT, FC_MODIFICACION)
SELECT DISTINCT UPPER(TRIM(BILL_CYCLE)) CICLO
, NOW(), NOW()
FROM STAGE.STG_FACTURAS_FCT FACT
WHERE TRIM(BILL_CYCLE)<>'';
INSERT INTO ODS.ODS_DM_CICLOS_FACTURACION VALUES(9999, 'DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS.ODS_DM_CICLOS_FACTURACION VALUES(9998, 'NO APLICA', NOW(), NOW());
COMMIT;

ANALYZE TABLE ODS.ODS_DM_CICLOS_FACTURACION;
```

 # 3.3 Se añaden los clientes de la tabla de STAGE del facturador que no existen en la tabla clientes en ODS

```
INSERT INTO ODS.ODS_HC_CLIENTES
(ID_CLIENTE
, NOMBRE_CLIENTE
, APELLIDOS_CLIENTE
, NUMDOC_CLIENTE
, ID_SEXO
, ID_DIRECCION_CLIENTE
, TELEFONO_CLIENTE
, EMAIL
, FC_NACIMIENTO
, ID_PROFESION
, ID_COMPANYA
, FC_INSERT
, FC_MODIFICACION)
SELECT DISTINCT(CUSTOMER_ID)
, 'DESCONOCIDO'
, 'DESCONOCIDO'
, '99-999-9999'
, 99
, 999999
, '9999999999'
, 'DESCONOCIDO'
, STR_TO_DATE('9999-12-31','%Y-%m-%d')
, 999
, 999
, NOW()
, NOW()
FROM STAGE.STG_FACTURAS_FCT SSFC
LEFT JOIN ODS.ODS_HC_CLIENTES OHCL
ON SSFC.CUSTOMER_ID = OHCL.ID_CLIENTE
WHERE OHCL.ID_CLIENTE IS NULL;
COMMIT;

ANALYZE TABLE ODS_HC_CLIENTES;
```

 # 3.4 Poblamos tabla hechos Facturas

```
INSERT INTO ODS.ODS_HC_FACTURAS
(ID_FACTURA
, ID_CLIENTE
, FC_INICIO
, FC_FIN
, FC_ESTADO
, FC_PAGO
, ID_CICLO_FACTURACION
, ID_METODO_PAGO
, CANTIDAD
, FC_INSERT
, FC_MODIFICACION)
SELECT SSFC.BILL_REF_NO
, OOHC.ID_CLIENTE
, CASE WHEN TRIM(SSFC.START_DATE)<>'' THEN STR_TO_DATE(LEFT(SSFC.START_DATE,10),'%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999','%d/%m/%Y') END FC_INICIO
, CASE WHEN TRIM(SSFC.END_DATE)<>'' THEN STR_TO_DATE(LEFT(SSFC.END_DATE,10),'%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999','%d/%m/%Y') END FC_FIN
, CASE WHEN TRIM(SSFC.STATEMENT_DATE)<>'' THEN STR_TO_DATE(LEFT(SSFC.STATEMENT_DATE,10),'%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999','%d/%m/%Y') END
FC_ESTADO
, CASE WHEN TRIM(SSFC.PAYMENT_DATE)<>'' THEN STR_TO_DATE(LEFT(SSFC.PAYMENT_DATE,10),'%Y-%m-%d') ELSE STR_TO_DATE('31/12/9999','%d/%m/%Y') END FC_PAGO
, OODC.ID_CICLO_FACTURACION
, OODM.ID_METODO_PAGO
, SSFC.AMOUNT
, NOW()
, NOW()
FROM STAGE.STG_FACTURAS_FCT SSFC
INNER JOIN ODS.ODS_HC_CLIENTES OOHC ON (OOHC.ID_CLIENTE=SSFC.CUSTOMER_ID)
INNER JOIN ODS.ODS_DM_CICLOS_FACTURACION OODC ON CASE WHEN TRIM(SSFC.BILL_CYCLE)<>'' THEN UPPER(TRIM(SSFC.BILL_CYCLE)) ELSE 'DESCONOCIDO'
END=OODC.DE_CICLO_FACTURACION
INNER JOIN ODS.ODS_DM_METODOS_PAGO OODM ON CASE WHEN TRIM(SSFC.BILL_METHOD)<>'' THEN UPPER(TRIM(SSFC.BILL_METHOD)) ELSE 'DESCONOCIDO'
END=OODM.DE_METODO_PAGO;
COMMIT;

ANALYZE TABLE ODS_HC_FACTURAS;
```

DESARROLLO MODELO LLAMADAS

 # 1) Creación de tablas en ODS

 # 1.1 Creación tabla dimensión departamentos

```
DROP TABLE IF EXISTS ODS.ODS_DM_DEPARTAMENTOS_CC;
CREATE TABLE ODS.ODS_DM_DEPARTAMENTOS_CC
(ID_DEPARTAMENTO_CC INT(10) UNSIGNED AUTO_INCREMENT PRIMARY KEY
, DE_DEPARTAMENTO_CC VARCHAR(512)
, FC_INSERT DATETIME
, FC_MODIFICACION DATETIME);

ANALYZE TABLE ODS.ODS_DM_DEPARTAMENTOS_CC;
```

 # 1.2 Creación tabla dimensión agentes

```
DROP TABLE IF EXISTS ODS.ODS_DM_AGENTES_CC;
CREATE TABLE ODS.ODS_DM_AGENTES_CC
(ID_AGENTE_CC INT(10) UNSIGNED AUTO_INCREMENT PRIMARY KEY
, DE_AGENTE_CC VARCHAR(512)
, FC_INSERT DATETIME
, FC_MODIFICACION DATETIME);

ANALYZE TABLE ODS.ODS_DM_AGENTES_CC;
```

 # 1.3 Creación tabla hechos llamadas

```
DROP TABLE IF EXISTS ODS.ODS_HC_LLAMADAS;
CREATE TABLE ODS.ODS_HC_LLAMADAS
(ID_LLAMADA INT(10) UNSIGNED AUTO_INCREMENT PRIMARY KEY
, ID_IVR INT(11)
, TELEFONO_LLAMADA BIGINT(20)
, FC_INICIO_LLAMADA DATETIME
, FC_FIN_LLAMADA DATETIME
, ID_DEPARTAMENTO_CC INT(10) UNSIGNED
, FLG_TRANSFERIDO BOOLEAN
, ID_AGENTE_CC INT(10) UNSIGNED
, FC_INSERT DATETIME
, FC_MODIFICACION DATETIME);

ANALYZE TABLE ODS.ODS_HC_LLAMADAS;
```

 # 2) Creación FKs

```
ALTER TABLE ODS.ODS_HC_LLAMADAS ADD INDEX fk_llamad_depart_idx (ID_DEPARTAMENTO_CC ASC);
ALTER TABLE ODS.ODS_HC_LLAMADAS ADD CONSTRAINT fk_llam_depart FOREIGN KEY(ID_DEPARTAMENTO_CC)
  REFERENCES ODS.ODS_DM_DEPARTAMENTOS_CC(ID_DEPARTAMENTO_CC);

ALTER TABLE ODS.ODS_HC_LLAMADAS ADD INDEX fk_llamad_agent_idx (ID_AGENTE_CC ASC);
ALTER TABLE ODS.ODS_HC_LLAMADAS ADD CONSTRAINT fk_llam_agent FOREIGN KEY(ID_AGENTE_CC)
  REFERENCES ODS.ODS_DM_AGENTES_CC(ID_AGENTE_CC);
```

3) Poblamos el modelo

3.1 Poblamos tabla dimensión departamentos

```
INSERT INTO ODS.ODS_DM_DEPARTAMENTOS_CC (DE_DEPARTAMENTO_CC, FC_INSERT, FC_MODIFICACION)
SELECT DISTINCT UPPER(TRIM(SERVICE)) DEPARTAMENTO_CC
, NOW(), NOW()
FROM STAGE.STG_CONTACTOS_IVR DEPARTAMENTO
WHERE TRIM(SERVICE)<>'';
INSERT INTO ODS.ODS_DM_DEPARTAMENTOS_CC VALUES (999, ‘DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS.ODS_DM_DEPARTAMENTOS_CC VALUES (998, 'NO APLICA', NOW(), NOW());
COMMIT;

ANALYZE TABLE ODS.ODS_DM_DEPARTAMENTOS_CC;
```

3.2 Poblamos tabla dimensión agentes

```
INSERT INTO ODS.ODS_DM_AGENTES_CC (DE_AGENTE_CC, FC_INSERT, FC_MODIFICACION)
SELECT DISTINCT UPPER(TRIM(AGENT)) DEPARTAMENTO_CC
, NOW(), NOW()
FROM STAGE.STG_CONTACTOS_IVR DEPARTAMENTO
WHERE TRIM(AGENT)<>'';
INSERT INTO ODS.ODS_DM_AGENTES_CC VALUES (9999, 'DESCONOCIDO', NOW(), NOW());
INSERT INTO ODS.ODS_DM_AGENTES_CC VALUES (9998, 'NO APLICA', NOW(), NOW());
COMMIT;

ANALYZE TABLE ODS.ODS_DM_AGENTES_CC;
```

 # 3.3 Poblamos tabla hechos llamadas

```
INSERT INTO ODS.ODS_HC_LLAMADAS
(ID_IVR
, TELEFONO_LLAMADA
, FC_INICIO_LLAMADA
, FC_FIN_LLAMADA
, ID_DEPARTAMENTO_CC
, FLG_TRANSFERIDO
, ID_AGENTE_CC
, FC_INSERT
, FC_MODIFICACION)
SELECT SCIV.ID
, CASE WHEN TRIM(SCIV.PHONE_NUMBER)<>'' THEN REPLACE(SCIV.PHONE_NUMBER,'-','') ELSE -1 END TELEFONO_LLAMADA
, CASE WHEN TRIM(SCIV.START_DATETIME)<>'' THEN STR_TO_DATE(LEFT(SCIV.START_DATETIME,19),'%Y-%m-%d %H:%i:%s') ELSE STR_TO_DATE('9999-12-31','%Y-%m-%d')
END FC_INICIO_LLAMADA
, CASE WHEN TRIM(SCIV.END_DATETIME)<>'' THEN STR_TO_DATE(LEFT(SCIV.END_DATETIME,19),'%Y-%m-%d %H:%i:%s') ELSE STR_TO_DATE('9999-12-31','%Y-%m-%d') END
FC_FIN_LLAMADA
, OODC.ID_DEPARTAMENTO_CC
, IF(UPPER(SCIV.FLG_TRANSFER)='FALSE', FALSE, TRUE) AS FLG_TRANSFERIDO
, OODA.ID_AGENTE_CC
, NOW()
, NOW()
FROM STAGE.STG_CONTACTOS_IVR SCIV
INNER JOIN ODS.ODS_DM_DEPARTAMENTOS_CC OODC ON CASE WHEN TRIM(SCIV.SERVICE)<>'' THEN UPPER(TRIM(SCIV.SERVICE)) ELSE 'DESCONOCIDO'
END=OODC.DE_DEPARTAMENTO_CC
INNER JOIN ODS.ODS_DM_AGENTES_CC OODA ON CASE WHEN TRIM(SCIV.AGENT)<>'' THEN UPPER(TRIM(SCIV.AGENT)) ELSE 'DESCONOCIDO' END=OODA.DE_AGENTE_CC;
COMMIT;

ANALYZE TABLE ODS.ODS_HC_FACTURAS;
```

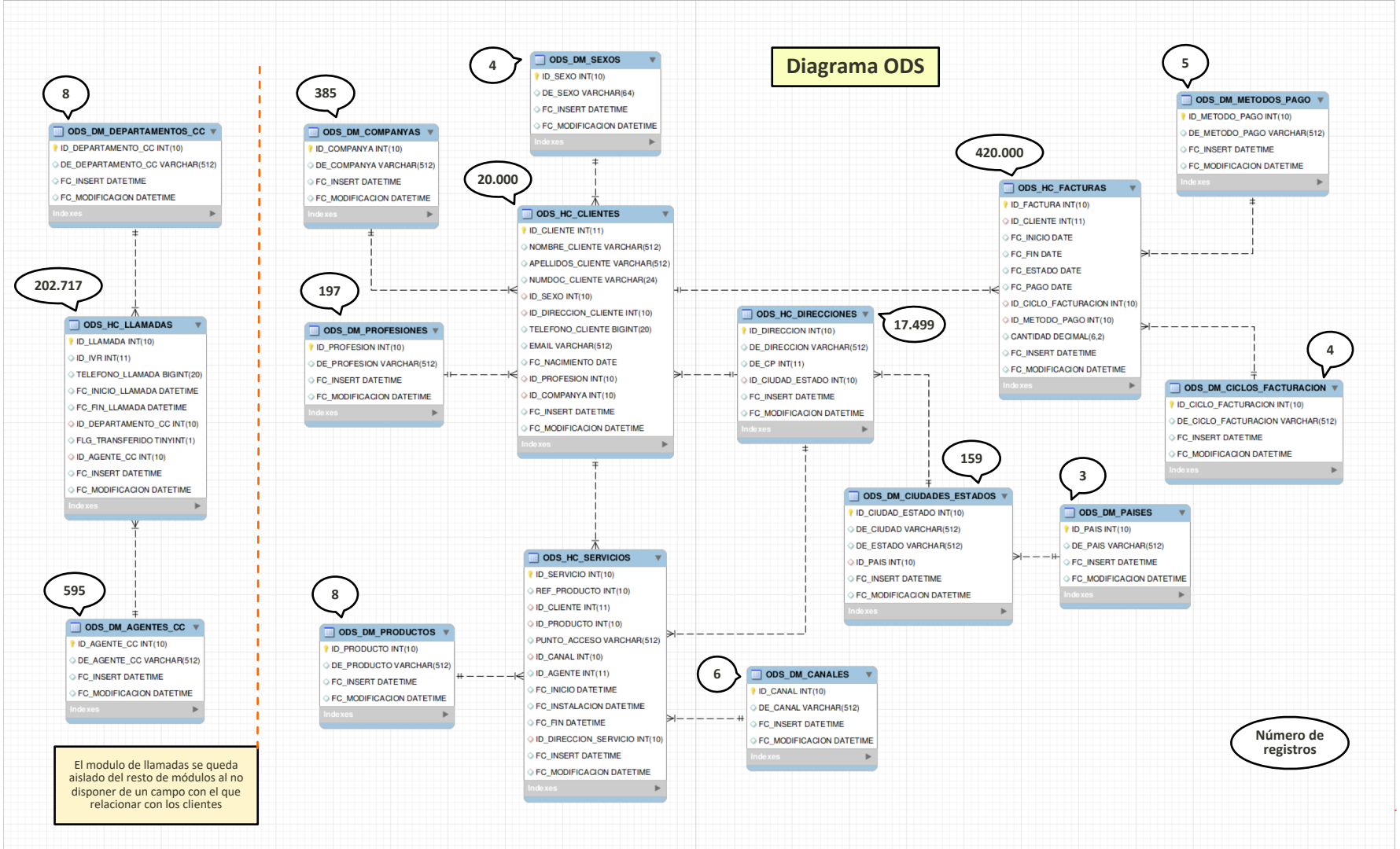


Diagrama ODS
8
ODS_DM_DEPARTAMENTOS_CC
ID_DEPARTAMENTO_CC INT(10)
DE_DEPARTAMENTO_CC VARCHAR(512)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
202.717
ODS_HC_LLAMADAS
ID_LLAMADA INT(10)
ID_IVR INT(11)
TELEFONO_LLAMADA BIGINT(20)
FC_INICIO_LLAMADA DATETIME
FC_FIN_LLAMADA DATETIME
ID_DEPARTAMENTO_CC INT(10)
FLG_TRANSFERIDO TINYINT(1)
ID_AGENTE_CC INT(10)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
595
ODS_DM_AGENTES_CC
ID_AGENTE_CC INT(10)
DE_AGENTE_CC VARCHAR(512)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
El modulo de llamadas se queda aislado del resto de módulos al no disponer de un campo con el que relacionar con los clientes
385
ODS_DM_COMPANYAS
ID_COMPANYA INT(10)
DE_COMPANYA VARCHAR(512)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
197
ODS_DM_PROFESIONES
ID_PROFESION INT(10)
DE_PROFESION VARCHAR(512)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
4
ODS_DM_SEXOS
ID_SEXO INT(10)
DE_SEXO VARCHAR(64)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
20.000
ODS_HC_CLIENTES
ID_CLIENTE INT(11)
NOMBRE_CLIENTE VARCHAR(512)
APELLIDOS_CLIENTE VARCHAR(512)
NUMDOC_CLIENTE VARCHAR(24)
ID_SEXO INT(10)
ID_DIRECCION_CLIENTE INT(10)
TELEFONO_CLIENTE BIGINT(20)
EMAIL VARCHAR(512)
FC_NACIMIENTO DATE
ID_PROFESION INT(10)
ID_COMPANYA INT(10)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
8
ODS_DM_PRODUCTOS
ID_PRODUCTO INT(10)
DE_PRODUCTO VARCHAR(512)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
ODS_HC_SERVICIOS
ID_SERVICIO INT(10)
REF_PRODUCTO INT(10)
ID_CLIENTE INT(11)
ID_PRODUCTO INT(10)
PUNTO_ACCESO VARCHAR(512)
ID_CANAL INT(10)
ID_AGENTE INT(11)
FC_INICIO DATETIME
FC_INSTALACION DATETIME
FC_FIN DATETIME
ID_DIRECCION_SERVICIO INT(10)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
6
ODS_DM_CANALES
ID_CANAL INT(10)
DE_CANAL VARCHAR(512)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
ODS_HC_DIRECCIONES
17.499
ID_DIRECCION INT(10)
DE_DIRECCION VARCHAR(512)
DE_CP INT(11)
ID_CIUDAD_ESTADO INT(10)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
159
ODS_DM_CIUDADES_ESTADOS
ID_CIUDAD_ESTADO INT(10)
DE_CIUDAD VARCHAR(512)
DE_ESTADO VARCHAR(512)
ID_PAIS INT(10)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
3
ODS_DM_PAISES
ID_PAIS INT(10)
DE_PAIS VARCHAR(512)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
420.000
ODS_HC_FACTURAS
ID_FACTURA INT(10)
ID_CLIENTE INT(11)
FC_INICIO DATE
FC_FIN DATE
FC_ESTADO DATE
FC_PAGO DATE
ID_CICLO_FACTURACION INT(10)
ID_METODO_PAGO INT(10)
CANTIDAD DECIMAL(6,2)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
5
ODS_DM_METODOS_PAGO
ID_METODO_PAGO INT(10)
DE_METODO_PAGO VARCHAR(512)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
4
ODS_DM_CICLOS_FACTURACION
ID_CICLO_FACTURACION INT(10)
DE_CICLO_FACTURACION VARCHAR(512)
FC_INSERT DATETIME
FC_MODIFICACION DATETIME
Indexes
Número de registros

➢ ¿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

Porque de esta forma se prioriza el rendimiento y velocidad, ya que permite indexar la dimensión de forma individualizada sin que repercuta en el rendimiento de la base de datos en su conjunto, a pesar de que al no estar normalizada pueda haber redundancia de datos y un ligero impacto en el espacio ocupado.

Aunque también entiendo que en función del análisis del alcance geográfico de los datos existentes, se podría incluir también el país dentro de la misma tabla.

➢ Separar el campo DE_DIRECCION de la tabla de direcciones en dos campos: NOMBRE_VIA y NUM_VIA

```
UPDATE ODS_HC_DIRECCIONES
SET NUM_VIA = SUBSTRING_INDEX(SUBSTRING_INDEX(DE_DIRECCION, ' ', 1), ' ', -1),
   NOMBRE_VIA = TRIM(SUBSTR(DE_DIRECCION, LOCATE(' ', DE_DIRECCION)))
WHERE ID_DIRECCION NOT IN (999998, 999999);

UPDATE ODS_HC_DIRECCIONES
SET NUM_VIA = 'DESCONOCIDO',
          NOMBRE_VIA = 'DESCONOCIDO'
WHERE ID_DIRECCION = 999999;

UPDATE ODS_HC_DIRECCIONES
SET NUM_VIA = 'NO APLICA',
          NOMBRE_VIA = 'NO APLICA'
WHERE ID_DIRECCION = 999998;
```

Explica qué habrías hecho diferente centrándote en las “patas”:

- Data Quality

  No es que lo hubiera hecho diferente, pero entiendo que centrándome en esta pata, es aquí donde habría de definirse las acciones a realizar para solventar los casos encontrados en los que la calidad de los datos no es la óptima:

  - Teléfonos de clientes no se pueden relacionar.
  - Facturas sobre las que no se puede identificar el cliente.

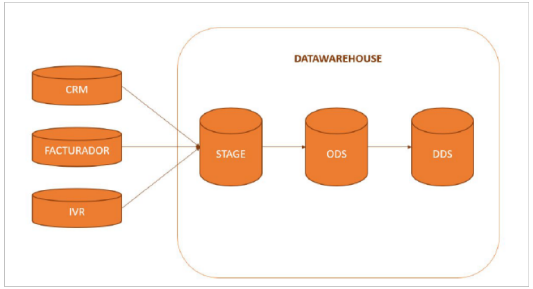
- Master Data

  Habría definido los datos (no se si sería correcto especificar dimensiones) que deberían ser estándar en la organización, como puede ser la tabla de países, para situarlos fuera del DataWarehouse y gestionar la traducción en la incorporación al DataWarehouse en caso necesario.

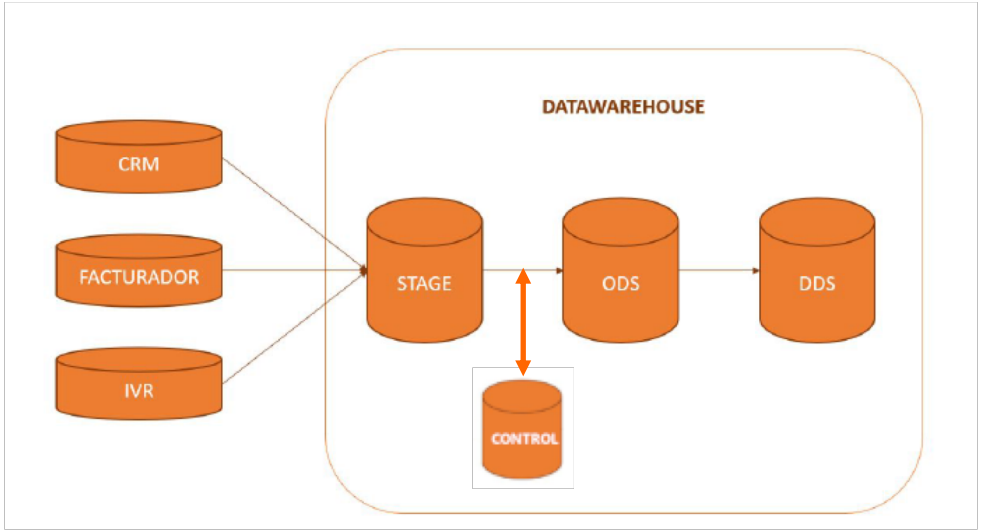
- Data Modeling & Design (me refiero a cómo están definidas las tablas en origen)

  Entendiendo que las tablas en origen no se pueden modificar, al pertenecer estas a los sistemas ya existentes en la organización (más allá de las acciones enumeradas en el punto anterior de Data Quality) creo que no haría nada diferente.

Después de todo lo visto nuestro ecosistema quedaría así:
¿Lo dejarías así o platearías otro diseño mejorado?



Tal vez incluiría un módulo con el que llevar un control de todas aquellas operaciones que se llevan a cabo para transformar la información en "bruto" al detalle que nos va a permitir analizar y tratar la información para adaptarnos a las necesidades de la empresa.



Escribe tus propias reglas o mandamientos de un DataWarehouse

1. **Un DataWarehouse se crea, no se compra (debe de estar adaptado a las necesidades de la empresa para ayudar en la toma de decisiones).**
2. **El diseño no es único ni estándar.**
3. **Ha de implantarse por fases.**
4. **La integridad de los datos siempre será el criterio de diseño más importante.**
5. **Siempre se debe considerar el rendimiento de la transacción en el diseño.**
6. **Los metadatos son un componente crítico de este entorno.**
7. **El proceso de extracción, transformación y carga (ETL), aunque tedioso, brinda la oportunidad de hacer que los datos sean más consistentes y precisos por lo que es una parte fundamental en la creación de un DataWarehouse.**
8. **Los datos han de ser seguros (control de vulnerabilidades).**
9. **Un DataWarehouse no es una base de datos relacional, no deben existir valores nulos y no tiene porque estar normalizada.**