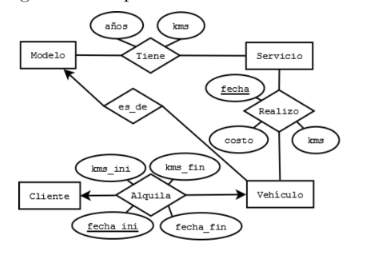
2)



Cliente (dni, nombre, apellido, fecha\_nac)

Modelo (marca, mod\_nombre, combustible)

Vehículo (patente, marca, mod\_nombre, fecha\_fabric, km\_actual)

Servicio (serv nombre, descripcion)

Tiene (marca, mod\_nombre, serv\_nombre, años, kms)

Realizo (patente, fecha, serv\_nombre, costo, kms)

Alquila (patente, fecha\_ini, kms\_ini, fecha fin, kms\_fin, dni)

a) A.R.: Vehículos a los cuales se les realizo un servicio que NO correspondía con su modelo (relación Tiene). Devuelva la patente del vehículo, marca, modelo, nombre del servicio, fecha y costo.

SERVICIOS\_REALIZADOS 🡨 ∏patente,nombre(Vehiculo JOIN Realizo)

SERVICIOS\_POSIBLES🡨 ∏patente,nombre(Vehiculo JOIN Tiene)

∏ patente, marca, mod\_nombre, serv\_nombre, fecha, costo((SERVICIOS\_REALIZADOS\SERVICIOS\_POSIBLES) JOIN REALIZO JOIN Vehiculo)

Esquema Vehiculo JOIN Realizo

(Patente, marca, mod\_nombre, fecha\_fabric, km\_actual, fecha, serv\_nombre, costo, ksm)

b) A.R. o SQL (a elección): DNI de los clientes que alquilaron un vehículo naftero (combustible "nafta") y luego alquilaron un vehículo gasolero (combustible="gasoil") de la misma marca. Puede utilizar los operadores relacionales (>, <,=,...) para comparar fechas. c) SQL:

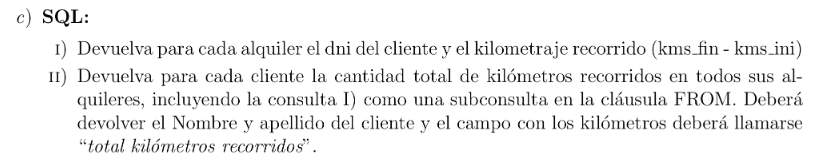
TEMP1🡨 JOIN Alquila JOIN Vehiculo

TEMP2🡨 JOIN Alquila JOIN Vehiculo

∏ dni(σ TEMP1.combustible = nafta (TEMP1))

∏ dni(σ TEMP1.combustible = gasolero (TEMP1))

Lo pasa javi



c) i)

SELECT dni, (kms\_fin-kms\_ini) as kms\_recorrido

FROM Alquila

INTO TEMP kilometraje // Aca le ponemos nombre a la consulta

c)ii)

SELECT dni, nombre, apellido

SUM kms\_recorrido as total\_kilómetros\_recorridos

FROM kilometraje NATURAL JOIN Cliente

GROUP BY dni, nombre, apellido (esto es xq queremos sumar respecto a las llaves dni, nombre y apellido)

INTO TEMP data\_cliente

SELECT nombre, apellido, total\_kilómetros\_recorridos

FROM data\_cliente

3) M = {CH->I, A-> CEF, BIJ->G, AH->I, B->AHI} R(ABCDEFGHIJ)

**1. Del lado derecho agregamos todo lo que podemos determinar con el lado izquierdo:**

CH+ = CHI A+=ACEF BIJ+= BIJGAHCEF AH+=AHICEF B+=BAHICEF

M1 = {CH 🡪 CHI, A 🡪 ACEF, BIJ 🡪 BIJGAHCEF, AH 🡪 AHICEF, B🡪 BAHICEF}

**2.Eliminamos dependencias redundantes**

CH 🡪 CHI es redundante?

CH+M1\{CH 🡪 CHI} = CH ⊉ CHI no es redundante

A+M1\{A🡪ACEF} = A ⊉ ACEF no es redundante

BIJ+M1\{BIJ🡪 BIJGAHCEF } = BIJAHICEF ⊉ BIJGAHCEF no es redundante

AH+M1\{ AH🡪AHICEF} = AHCEFIBJ ⊉ AHICEF es redundante

B+M1\{ B🡪 BAHICEF} = B ⊉ BAHICEF no es redundante

M2 = { CH 🡪 CHI, A 🡪 ACEF, BIJ 🡪 BIJGAHCEF, B🡪 BAHICEF}

**3.Reduzca a izquierda:**

Eliminamos atributos extraños a izquierda, sea una df XA->Y, A es extraño a izquierda si podemos generar X->Y.

A 🡪 ACEF, B🡪 BAHICEF como tienen **un solo atrib a izquierda** no hay que reducir a izq.

Veamos CH🡪CHI

¿H es extraño a izquierda?

C+M2= C ⊉ CHI por lo tanto H no es extraño a izq

¿C es extraño a izquierda?

H+M2 = H ⊉ CHI por lo tanto C no es extraño a izq

Veamos BIJ🡪BIJGAHCEF

¿J es extraño a izq?

BI+M2 = BIAHCEF ⊉ BIJGAHCEF no es extraño a izq

¿B es extraño a izq?

IJ+M2 = IJ ⊉ BIJGAHCEF no es extraño a izquierda

¿I es extraño a izq?

BJ+M2 = BJAHICEF ⊇ BIJGAHCEF por lo tanto I es extraño a izq

M3 = { CH 🡪 CHI, A 🡪 ACEF, BJ 🡪 BIJGAHCEF, B🡪 BAHICEF}

**4.Reduzca a derecha**

**Eliminamos triviales:**

M4 = { CH 🡪 I, A 🡪 CEF, BJ 🡪 IGAHCEF, B🡪 AHICEF}

En A🡪CEF

¿C es extraño a der?

A+\M4\{A🡪CEF}u{A🡪EF} = AEF ⊉ CEF , C no es extr a der

¿E es extraño a der?

A+\M4\{A🡪CEF}u{A🡪CF} = ACF ⊉ CEF , E no es extr a der

¿F es extraño a der?

A+\M4\{A🡪CEF}u{A🡪CE} = ACE ⊉ CEF , F no es extr a der

En BJ 🡪 IGAHCEF

¿I es extra a der?

BJ+M4\{ BJ 🡪 IGAHCEF}u{ BJ 🡪 GAHCEF } = BJGAHCEFI, I es extr a der

M5 = { CH 🡪 I, A 🡪 CEF, BJ 🡪 GAHCEF, B🡪 AHCEF} **LISTORTI**

5) F={D🡪B, I🡪F, E🡪AC, F🡪HIJ, EH🡪DF, ABC🡪DE} conj R(ABCDEFGHIJ)

Llaves candidatas: ABCF, ABCH, ABCI, ACDF, ACDH, ACDI, EF, EH, EI

Encontrar 3FN:

**1.Abro a derecha**

M1={D🡪B, I🡪F, E🡪A, E🡪C, F🡪H,F🡪I,F🡪J,EH🡪D,EH🡪F,ABC🡪D,ABC🡪E}

Reordenamos para que los que tengan más atrib a izquierda queden a la izquierda:

M2={ ABC🡪D,ABC🡪E, EH🡪D,EH🡪F, D🡪B, I🡪F, E🡪A, E🡪C, F🡪H,F🡪I,F🡪J}

**2. Verificamos si R esta en 3era forma normal** (3FN), o sea, si en cada dependencia funcional el **antecedente es super llave** **y** el **consecuente** **es** **parte** de **alguna** **llave** (es primo).

ABC🡪D como el antecendente no es super llave entonces no está en 3FN.

M2={ ABC🡪D,ABC🡪E, EH🡪D,EH🡪F, D🡪B, I🡪F, E🡪A, E🡪C, F🡪H,F🡪I,F🡪J}

**3.** Si F no está en 3FN entonces **hay** **que** **formar** **subesquemas** por **cada** **dependencia** **funcional**, **calculando** todas las df que se proyectan en cada uno.

Evaluamos: Que todas las letras del conjunto del subesquema se encuentren a izquierda en M2, si es asi hay q evaluar todas las letras.

Creamos un subesquema para la df ABC🡪D:  
Subesquema 1: conjunto (ABCD)

Calculamos las llaves de ABC🡪D

A+M2=A no se deduce ninguna df no trivial

B+M2 = B no se deduce ninguna df no trivial

C+M2 = C “ “

D+M2 = DB podemos deducir D🡪B, se proyecta sobre ABCD

Lo incorporamos al nuevo subesquema: {ABC🡪D, D🡪B}

Combinaciones 2 atributaciones:

AB+M2= AB no deduce nada

AC+M2 = AC no deduce

AD+M2 = AD no deduce nada

BC+M2 = BC no deduce nada

BD+M2 = BD no deduce nada

CD+M2 = CD no deduce nada

Combinaciones de 3 atributos

ABC+M2 = ABCDE no deduce nada (es trivial)

ABD+M2 = ABD no deduce D porque D es parte de la identidad.

ACD+M2 = ACDBE se deduce B pero es redundante con D🡪B

BCD+M2 = BCD

Luego el subesquema 1: (ABCD) {ABC🡪D, D🡪B} calcular llaves

Siempre: AC

Talvez: B,D

AC+S1 = AC

Combinaciones 3 atrib:

ACB+S1 = ACBD ⊇ S1

ACD+S1 = ACDB ⊇ S1

Combinaciones 4 atrib: No tiene sentido seguir

(ABCD) {ABC🡪D, D🡪B} llaves = ACB, ACD

M2={ ABC🡪D,ABC🡪E, EH🡪D,EH🡪F, D🡪B, I🡪F, E🡪A, E🡪C, F🡪H,F🡪I,F🡪J}

Subesquema 2: Conj ABCE

Izq(ABC) = ABCE

ABC🡪E

Calculamos como si fuesen llaves:

A+M2=A

B+M2=B

C+M2=C

E+M2=EAC con la E se deduce E🡪AC, como AC se proyecta ABCE entonces las df quedan asi:  
{ABC🡪E, E🡪AC}

Combinaciones 2 atributos:

AB+M2= AB

AC+M2=AC

AE+M2=AEC, se deduce AE🡪C se agrega {ABC🡪E, E🡪AC, AE🡪C}

BC+M2=BC

BE+M2=BEACD , BE🡪AC, como ya tenemos E🡪AC es redundante agregarlo

CE+M2=CEA, se deduce CE🡪A se agrega {ABC🡪E, E🡪AC, AE🡪C,CE🡪A}

Combinaciones de 3 atrib:

ABC es trivial

ABE+M2= ABECD se ABE🡪C redundante

ACE+M3=ACE

BCE+M2= BCEAD BCE🡪A redundante

Combinaciones de 4 atrib: Listorti

Calculamos las llaves para S2(ABCE) {ABC🡪E, E🡪AC, AE🡪C,CE🡪A}

Siempre: B

Talvez: ACE

Izq(ABCE)=B

Calculo llaves:

B+=B

Combinaciones de 2 atrib:

BA+=BA

BC+=BC

BE+=BEAC es llave

Combinaciones de 3 atrib:

BAC+=BACE es llave

BAE+=BAEC es super llave de BE

Combinaciones de a 4 no tienen sentido

S2(ABCE) {ABC🡪E, E🡪AC, AE🡪C,CE🡪A} llaves: BAC, BAE