

Ficheros y Bases de Datos



Práctica 2

Grupo 82

Laura Álvarez Flórez
Laura Yunta García

100363965
100363785

100363965@alumnos.uc3m.es
100363785@alumnos.uc3m.es

Profesor:	Francisco Javier Calle Gómez	Grupo	82
Alumno/a:	Laura Álvarez Florez	NIA:	100363965
Alumno/a:	Laura Yunta García	NIA:	100363785

1. Introducción

El problema propuesto por la práctica tratará de resolver las distintas consultas, funciones, procedimientos, el diseño externo y los triggers que utiliza el sistema desarrollado en la práctica anteriormente resulta cuya función era buscar un sistema que pueda tramitar las sanciones a vehículos que han sido observados mediante radares cometiendo alguna infracción en su tránsito por algunas carreteras.

La resolución vendrá estructurada en siete subapartados:

1. En el cual se resuelven las consultas dadas por el enunciado en algebra relacional e implementarlas en SQL.
2. En el cual se desarrollan las funciones dadas por el enunciado.
3. En el cual se crean el procedimiento pedido por el enunciado.
4. En el cual se crean de las vistas solicitadas por el enunciado de la práctica.
5. En el cual se desarrolla el diseño externo solicitado
6. En el cual se crearán los disparadores pedidos por el enunciado cuya función será incorporar la semántica.
7. Conclusiones finales y notas sobre el desarrollo de la práctica


Para cada subapartado se facilita una tabla donde se han incorporado los diferentes aspectos a comentar como nombre, implementación, diseño de la estructura, algebra relacional o pruebas realizadas.

Finalmente con el contenido de este documento final se busca la recolección de datos, pruebas e implementaciones para el desarrollo de la solución del problema, además en este documento están determinadas las decisiones de diseño tomadas durante dicho proceso de desarrollo y especificadas todas y cada una de las pruebas que se ha decidido realizar para asegurar la eficiencia del resultado.

2. Consultas

Consulta 1																							
Descripción	Los 10 vehículos más 'observados' en el transcurso del día de hoy.																						
Algebra relacional	$\sigma_{\text{rownum} < 11} \Pi_{\text{nplate}, \rho_{\text{contador}}(\text{Count}(\text{nPlate}))} \sigma_{\text{TRUNC}(\text{odatetime}) = \text{TRUNC}(\text{SYSTIMESTAMP})} \text{G}_{\text{nPlate}} \text{T}_{\text{contador}}(\text{Observations})$																						
SQL	<pre> SELECT * FROM (SELECT nPlate, COUNT(nPlate) as contador FROM observations WHERE TRUNC(odatetime) = TRUNC(SYSTIMESTAMP) GROUP BY nPlate ORDER BY contador DESC) WHERE rownum < 11; </pre>																						
Pruebas realizadas	<pre> SQL> SELECT * FROM (2 SELECT nPlate, COUNT(nPlate) as contador 3 FROM observations 4 WHERE TRUNC(odatetime) = TRUNC(to_timestamp('1-jul-10 07.00.00 PM')) 5 GROUP BY nPlate 6 ORDER BY contador DESC 7) 8 WHERE rownum < 11; </pre> <table> <thead> <tr> <th>NPLATE</th><th>CONTADOR</th></tr> </thead> <tbody> <tr><td>5137IU0</td><td>2</td></tr> <tr><td>5408E11</td><td>2</td></tr> <tr><td>5755011</td><td>2</td></tr> <tr><td>7250EOE</td><td>2</td></tr> <tr><td>3422AEU</td><td>2</td></tr> <tr><td>3812E1U</td><td>2</td></tr> <tr><td>682910A</td><td>2</td></tr> <tr><td>8960AEO</td><td>2</td></tr> <tr><td>5938A01</td><td>2</td></tr> <tr><td>6817EUA</td><td>2</td></tr> </tbody> </table> <p>10 rows selected.</p>	NPLATE	CONTADOR	5137IU0	2	5408E11	2	5755011	2	7250EOE	2	3422AEU	2	3812E1U	2	682910A	2	8960AEO	2	5938A01	2	6817EUA	2
NPLATE	CONTADOR																						
5137IU0	2																						
5408E11	2																						
5755011	2																						
7250EOE	2																						
3422AEU	2																						
3812E1U	2																						
682910A	2																						
8960AEO	2																						
5938A01	2																						
6817EUA	2																						

Consulta 2	
Descripción	Listado de carreteras y su valor de velocidad promedio establecida, ordenado de mayor a menor velocidad en primera instancia y por orden alfabético de carreteras en segunda, contando ambos sentidos.
Algebra relacional	$\Pi_{\text{owner}, \rho_{\text{veloc_promed}}(\text{AVG}(\text{speedlim}))} \text{G}_{\text{road}} \text{T}_{(\text{Veloc_promed}), \text{road}}(\text{radars})$
SQL	<pre> SELECT road, AVG(speedlim) as veloc_promed FROM radars GROUP BY road ORDER BY veloc_promed DESC, road; </pre>

Titulación: GRADO INGENIERIA INFORMATICA Año Académico: 2017/2018 Curso: 2º Asignatura: Ficheros y Bases de Datos Título: Memoria Práctica 2 – Consultas, vistas y disparadores	 uc3m Universidad Carlos III de Madrid
---	--

Pruebas realizadas	<pre>SQL> SELECT road, AVG(speedlim) as veloc_promed 2 FROM radars 3 GROUP BY road 4 ORDER BY veloc_promed DESC, road;</pre> <pre>ROAD VELOC_PROMED ----- A1 100 A2 100 A3 100 A4 100 A5 100 A6 100 M50 100 M40 80 M45 80 M30 50 10 rows selected.</pre>
--------------------	--

Consulta 3	
Descripción	Personas que no conducen ninguno de sus vehículos (ni como conductor habitual ni como conductor adicional).
Algebra relacional	$\Pi_{\text{owner}} \neg \text{owner} (\Pi_{\text{owner}, \text{reg_driver}} \sigma_{\text{owner} \neq \text{reg_driver} \text{ AND } \Pi_{\text{Count}(\text{driver})} \sigma_{\text{assignments.driver}=\text{vehicles.owner AND assignments.nPlate}=\text{vehicles.nPlate}} (\text{assignments}) < 1} (\text{vehicles}))$
SQL	<pre>SELECT DISTINCT owner, FROM (SELECT owner, reg_driver FROM vehicles WHERE owner != reg_driver AND (SELECT COUNT(driver) FROM assignments WHERE assignments.driver = vehicles.owner AND assignments.nPlate = vehicles.nPlate) < 1) ORDER BY owner;</pre>

Pruebas realizadas	<pre>SQL> SELECT DISTINCT owner 2 FROM (3 SELECT owner, reg_driver 4 FROM vehicles 5 WHERE owner != reg_driver AND 6 (SELECT COUNT(driver) 7 FROM assignments 8 WHERE assignments.driver = vehicles.owner 9) <1 10) 11 ORDER BY owner;</pre> <p>OWNER ----- 00015430C (-----) OWNER ----- 86515357Y 90553454Q 92366576W 92453827Z 93655750A 97045896H 97201505D 97959442G 99181271G 97 rows selected.</p>	<pre>SQL> INSERT INTO assignments VALUES('99181271G', '64930EA'); 1 row created.</pre> <p>OWNER ----- 86515357Y 90553454Q 92366576W 92453827Z 93655750A 97045896H 97201505D 97959442G 99181271G 96 rows selected.</p> <p>Para comprobar el correcto funcionamiento añadimos una asignación, resultado esperado: un owner menos en la lista.</p>
--------------------	--	--


Consulta 4	
Descripción	Jefazo: dueños de al menos tres coches que no son conductores.
Algebra relacional	$\Pi_{owner} \sigma_{owner \geq 3} \neg c_owner (\Pi_{owner, pc_owner} (Count(owner)) \sigma_{owner \neq reg_driver} (owner)(Vehicles))$
SQL	<pre>SELECT owner FROM (SELECT owner, COUNT(owner) AS c_owner FROM vehicles WHERE owner != reg_driver GROUP BY owner) WHERE c_owner >= 3 ORDER BY c_owner DESC;</pre>
Pruebas realizadas	<pre>SQL> SELECT owner 2 FROM (3 SELECT owner, COUNT(owner) AS c_owner 4 FROM vehicles 5 WHERE owner != reg_driver 6 GROUP BY owner 7) 8 WHERE c_owner >= 3 9 ORDER BY c_owner DESC;</pre> <p>OWNER ----- 05511330R 56651407S 22649968M 82883718K 48906593Z 17017996C 19425545K 49806223E 60908146Y 48272209Q 10 rows selected.</p>

Consulta 5	
Descripción	Evolución: indica la diferencia de ingresos por multas entre el mes pasado y el mismo mes del año anterior.
Algebra relacional	$\rho_{\text{diferencia}} (\Pi(\Pi_{\text{SUM}(\text{amount})} \sigma \text{EXTRACT}(\text{YEAR FROM pay_date}) = \text{EXTRACT}(\text{YEAR FROM SYSDATE}) \text{ AND } \text{EXTRACT}(\text{MONTH FROM pay_date}) = \text{EXTRACT}(\text{MONTH FROM (ADD MONTHS (SYSDATE,-1))}(\text{tickets})) - (\Pi_{\text{SUM}(\text{amount})} \sigma \text{EXTRACT}(\text{YEAR FROM pay_date}) = \text{EXTRACT}(\text{YEAR FROM SYSDATE})-1 \text{ AND } \text{EXTRACT}(\text{MONTH FROM pay_date}) = \text{EXTRACT}(\text{MONTH FROM (ADD MONTHS (SYSDATE,-1))}(\text{tickets}))(DUAL))$
SQL	<pre> SELECT (SELECT SUM(amount) FROM tickets WHERE EXTRACT(YEAR FROM pay_date) = EXTRACT(YEAR FROM SYSDATE) AND EXTRACT(MONTH FROM pay_date) = EXTRACT(MONTH FROM (ADD_MONTHS (SYSDATE, -1)))) - (SELECT SUM(amount) FROM tickets WHERE EXTRACT(YEAR FROM pay_date) = EXTRACT(YEAR FROM SYSDATE)-1 AND EXTRACT(MONTH FROM pay_date) = EXTRACT(MONTH FROM (ADD_MONTHS (SYSDATE, -1)))) AS diferencia FROM DUAL; </pre>
Pruebas realizadas	<pre> SQL> select obs1_date, amount from tickets; OBS1_DATE AMOUNT ----- 11-MAR-18 09.48.00.000000 AM 300 12-MAR-18 09.48.00.000000 AM 400 11-MAR-17 09.48.00.000000 AM 150 12-MAR-17 09.48.00.000000 AM 50 SQL> SELECT 2 (3 SELECT SUM(amount) 4 FROM tickets 5 WHERE EXTRACT(YEAR FROM pay_date) = EXTRACT(YEAR FROM SYSDATE) 6 AND EXTRACT(MONTH FROM pay_date) = EXTRACT(MONTH FROM (ADD_MONTHS(SYSDATE,-1))) 7) - 8 (9 SELECT SUM(amount) 10 FROM tickets 11 WHERE EXTRACT(YEAR FROM pay_date) = EXTRACT(YEAR FROM SYSDATE)-1 12 AND EXTRACT(MONTH FROM pay_date) = EXTRACT(MONTH FROM (ADD_MONTHS(SYSDATE,-1))) 13) AS diferencia 14 FROM DUAL; DIFERENCIA ----- 500 </pre> <p>Se han realizado las inserciones mostradas en la captura en la tabla alegaciones y el resultado es el esperado = 500.</p>

3. Paquete


Paquete	
Implementación SQL	<pre> CREATE OR REPLACE PACKAGE package_functions AS -- Variables acceso externo a este package --Funciones acceso externo a este package FUNCTION AMOUNT_R(VEL_VEH NUMBER, VEL_RAD NUMBER) RETURN NUMBER; FUNCTION AMOUNT_T(RAD1_PK NUMBER, RAD1_TIME TIMESTAMP, RAD2_PK NUMBER, RAD2_TIME TIMESTAMP, VEL_ROAD NUMBER) RETURN NUMBER; FUNCTION AMOUNT_D(TS1 TIMESTAMP, TS2 TIMESTAMP) RETURN NUMBER; FUNCTION O_BEF_RADAR(P_DATETIME TIMESTAMP, P ROAD VARCHAR2, P_KM_POINT NUMBER, P_DIRECTION VARCHAR2) RETURN OBS_TYPE; FUNCTION O_BEF_VEHICULO(P_PLATE VARCHAR2, P_DATETIME TIMESTAMP) RETURN OBS_TYPE; END package_functions; / </pre>

Funcion 1	AMOUNT_R
Descripción	Cuantía para una sanción de velocidad máxima de radar.
Cálculo	<p>Esta función resta las dos velocidades que le son introducidas por parámetro, a este resultado se le aplica una función “techo” y se multiplica por 10 para obtener la cuantía a pagar.</p> <p>Se ha tenido en cuenta que el resultado podría ser negativo si la velocidad del vehículo no superase la del radar.</p>
Implementación SQL	<pre> CREATE ORREPLACE PACKAGE BODY package_functions AS FUNCTION AMOUNT_R(VEL_VEH NUMBER, VEL_RAD NUMBER) RETURN NUMBER IS RESULTADO NUMBER; BEGIN RESULTADO := VEL_VEH - VEL_RAD; RESULTADO := CEIL(RESULTADO); IF RESULTADO < 0 THEN RESULTADO := 0; END IF; RESULTADO := RESULTADO*10; RETURN RESULTADO; END; </pre>
Prueba	<pre> SQL> SELECT package_functions.amount_r(150, 120) FROM DUAL; PACKAGE_FUNCTIONS.AMOUNT_R(150,120) ----- 300 </pre>

Titulación: GRADO INGENIERIA INFORMATICA Año Académico: 2017/2018 Curso: 2º Asignatura: Ficheros y Bases de Datos Título: Memoria Práctica 2 – Consultas, vistas y disparadores	 uc3m Universidad Carlos III de Madrid
---	--

Funcion 2	AMOUNT_T
Descripción	Cuantía para una sanción de velocidad de tramo.
Cálculo	<p>La función amount_t resta los puntos km de las dos observaciones y las fechas de ambas, calcula la velocidad media del vehiculo y haciendo una llamada a amount_r se calcula la cuantia asociada a dicha velocidad.</p> <p>Se ha controlado que la división no se realice por el numero 0.</p>
SQL	<pre> FUNCTION AMOUNT_T(RAD1_PK NUMBER, RAD1_TIME TIMESTAMP, RAD2_PK NUMBER, RAD2_TIME TIMESTAMP, VEL_ROAD NUMBER) RETURN NUMBER IS RESULTADO NUMBER; VEL_VEH NUMBER; AUX1 NUMBER; AUX2 NUMBER; BEGIN AUX1 := (RAD2_PK) - (RAD1_PK); AUX2 := (CAST(RAD2_TIME AS DATE) - CAST(RAD1_TIME AS DATE)) *24; RESULTADO :=0; If AUX2 != 0 THEN VEL_VEH := AUX1/AUX2; RESULTADO := AMOUNT_R (VEL_VEH, VEL_ROAD); END IF; RETURN RESULTADO; END;</pre>
Prueba	<pre> SQL> SELECT package_functions.amount_t(0, TO_TIMESTAMP('17-APR-18 5.08.46.75'), 140, TO_TIMESTAMP(' 7-APR-18 6.08.46.75'), 120) FROM DUAL; PACKAGE_FUNCTIONS.AMOUNT_T(0,TO_TIMESTAMP('17-APR-185.08.46.75'),140,TO_TIMESTAMP ----- 200</pre> <p>En el primer caso el resultado esperado según el calculo concuerda. En el caso de introducir el mismo valor en la fecha y hora de ambas observaciones de radar, resultado esperado 0;</p> <pre> AMOUNT_T(0,TO_TIMESTAMP('18-MAR-181.48.00.00.AM'),140,TO_TIMESTAMP('18-MAR-181.4 ----- 0</pre>


Funcion 3	AMOUNT_D
Descripción	Cuantía para una sanción de distancia.
Cálculo	<p>Teniendo dos distancias se calcula la diferencia en segundos de ambas, teniendo en cuenta la posibilidad de un numero negativo y a continuación se calcula la cuantia asociada siendo el numero convertido a decimas de segundo y luego multiplicado por 10E por decima.</p>
Implementación SQL	<pre> FUNCTION AMOUNT_D(TS1 TIMESTAMP, TS2 TIMESTAMP) RETURN NUMBER IS RESULTADO NUMBER; SEGUNDOS1 NUMBER; SEGUNDOS2 NUMBER; BEGIN SEGUNDOS1 := EXTRACT(SECOND FROM TS1); SEGUNDOS2 := EXTRACT(SECOND FROM TS2); RESULTADO := SEGUNDOS1 - SEGUNDOS2; RESULTADO := 3.6 - ABS(RESULTADO); IF RESULTADO < 0 THEN RESULTADO := 0; END IF; RESULTADO := RESULTADO*10*10;</pre>

Titulación: GRADO INGENIERIA INFORMATICA Año Académico: 2017/2018 Curso: 2º Asignatura: Ficheros y Bases de Datos Título: Memoria Práctica 2 – Consultas, vistas y disparadores	 uc3m Universidad Carlos III de Madrid
---	--


	RETURN RESULTADO; END;
Prueba	<pre>SQL> SELECT package_functions.AMOUNT_D('27-JAN- 12 2 12.04.32.7 AM', '27-JAN- 12 3 12.04.35.7 AM') FROM DUAL; PACKAGE_FUNCTIONS.AMOUNT_D('27-JAN-1212.04.32.7AM', '27-JAN-1212.04.35.7AM') ----- 60</pre>

Funcion 4	O_BEF_RADAR
Descripción	Observación inmediatamente anterior a otra observación (del mismo radar).
Algebra relacional	$\Pi_{odateime, nPlate \sigma P_DATETIME > odateime \text{ AND } P_ROAD = road \text{ AND } P_KM_POINT = km_point \text{ AND } P_DIRECTION = direction}$ (OBSERVATIONS)
Implementación SQL	<pre>FUNCTION O_BEF_RADAR(P_DATETIME TIMESTAMP, P_ROAD VARCHAR2, P_KM_POINT NUMBER, P_DIRECTION VARCHAR2) RETURN OBS_TYPE IS MIN_DATETIME TIMESTAMP; PLATE_REC VARCHAR2(7); CURSOR c1 IS SELECT odateime, nPlate FROM OBSERVATIONS WHERE P_DATETIME > odateime AND P_ROAD = road AND P_KM_POINT = km_point AND P_DIRECTION = direction; BEGIN MIN_DATETIME := '1-JAN-00 01.01.32.700000 AM'; PLATE_REC := ''; FOR rec IN c1 LOOP IF rec.odateime > MIN_DATETIME THEN MIN_DATETIME := rec.odateime; PLATE_REC := rec.nPlate; END IF; END LOOP; RETURN OBS_TYPE(PLATE_REC, MIN_DATETIME); END;</pre>
	<pre>CREATE OR REPLACE TYPE OBS_TYPE AS OBJECT (P_NPLATE VARCHAR2(7), P_ODATETIME TIMESTAMP);</pre>
Prueba	<pre>SQL> SELECT package_functions.O_BEF_RADAR('27-JAN-12 12.04.32.700000 AM', 'A1', 76, 'ASC') FROM DUAL; PACKAGE_FUNCTIONS.O_BEF_RADAR('27-JAN-1212.04.32.700000AM', 'A1', 76, 'ASC')(P_NPLA OBS_TYPE('1556IAI', '20-JAN-12 10.37.33.570000 AM'))</pre>

Funcion 5	O_BEF_VEHICULO
Descripción	Observación inmediatamente anterior a otra observación (del mismo vehículo)
Algebra relacional	$\Pi_{odateime, nPlate \sigma P_DATETIME > odateime \text{ AND } P_PLATE = nplate}$ (OBSERVATIONS)
SQL	<pre>FUNCTION O_BEF_VEHICULO(P_PLATE VARCHAR2, P_DATETIME TIMESTAMP)</pre>

Titulación: GRADO INGENIERIA INFORMATICA Año Académico: 2017/2018 Curso: 2º Asignatura: Ficheros y Bases de Datos Título: Memoria Práctica 2 – Consultas, vistas y disparadores	 uc3m Universidad Carlos III de Madrid
---	--

	<pre> RETURN OBS_TYPE IS MIN_DATETIME TIMESTAMP; PLATE_REC VARCHAR2(7); CURSOR c1 IS SELECT odatetime, nPlate FROM OBSERVATIONS WHERE P_DATETIME > odatetime AND P_PLATE = nPlate; BEGIN MIN_DATETIME := '1-JAN-00 01.01.32.700000 AM'; PLATE_REC := ''; FOR rec IN c1 LOOP IF rec.odatetime > MIN_DATETIME THEN MIN_DATETIME := rec.odatetime; PLATE_REC := rec.nPlate; END IF; END LOOP; RETURN OBS_TYPE(PLATE_REC,MIN_DATETIME); END; END package_functions; </pre>
	<pre> --Estructura de tipo obs_type que se devuelve por la funcion CREATE OR REPLACE TYPE OBS_TYPE AS OBJECT (P NPLATE VARCHAR2(7), P ODATETIME TIMESTAMP); </pre>
Prueba	<pre> SQL> SELECT 2 Package_functions.O_BEF_VEICULO ('12860E0','27-JAN-12 01.15.42.320000 AM') FROM DUAL; PACKAGE_FUNCTIONS.O_BEF_VEICULO('12860E0','27-JAN-1201.15.42.320000AM')(P_NPLAT ----- OBS_TYPE('12860E0', '27-JAN-12 12.04.32.700000 AM') </pre>

Titulación: GRADO INGENIERIA INFORMATICA Año Académico: 2017/2018 Curso: 2º Asignatura: Ficheros y Bases de Datos Título: Memoria Práctica 2 – Consultas, vistas y disparadores	 uc3m Universidad Carlos III de Madrid
---	--


4. Vistas

Vista 1	Nueva_Multa
Descripción	Vista que proporciona los vehículos infractores de una velocidad anormalmente reducida, la fecha, y la diferencia de velocidad con el margen legal.
Algebra relacional	$\Pi_{\text{observations.nPlate, observations.odatetime, observations.speed, roads.speed_limit}} \sigma_{\text{observations.speed} < (\text{roads.speed_limit}/2)} \bowtie_{\text{observations.road=roads.name}} (\text{observations} \bowtie \text{roads})$
SQL	<pre>CREATE OR REPLACE VIEW Nueva_Multa AS SELECT observations.nPlate, observations.odatetime, observations.speed, roads.speed_limit FROM observations INNER JOIN roads ON observations.road = roads.name WHERE observations.speed < (roads.speed_limit / 2) ORDER BY nPlate, odatetime;</pre>
Funcionalidad	Inserción. Esta vista facilita los datos de los individuos infractores a los que se les podría insertar una multa.
Pruebas	<pre>SELECT * FROM nueva_multa; Se muestra el final del resultado 9834EU0 NPLATE ----- ODATETIME ----- SPEED SPEED_LIMIT ----- 16-MAY-10 01.14.16.310000 AM 33 70 34 rows selected.</pre>

Vista 2	Protestón
Descripción	Conductores con más alegaciones rechazadas para cada mes.
Algebra relacional	$\Pi_{\text{year_p.month_p, dr_proteston, ptimes (Max(cuenta))}} \Join_{\text{year_p, month_p, dr_proteston}} \bowtie_{\text{year_p, month_p}} (\Pi_{\text{year_p}} (\text{EXTRACT(YEAR FROM A.reg_date)}),$ $\rho_{\text{month_p}} (\text{EXTRACT(MONTH FROM A.reg_date)}), \rho_B.\text{debtor} (\text{dr_proteston}),$ $\rho_{\text{cuenta}} (\text{Count(B.debtor)}) \Join_{\text{EXTRACT(YEAR FROM A.reg_date), EXTRACT(MONTH FROM A.reg_date), B.debtor}} \bowtie_{\text{year_p, month_p, cuenta}} (\text{allegations A} \bowtie_{\text{A.obs_veh=B.obs_veh}} \text{tickets B}))$ $\text{AND A.obs_date=B.obs_date AND A.tik_type=B.tik_type}$
SQL	<pre>CREATE OR REPLACE VIEW proteston AS SELECT year_p, month_p, dr_proteston, MAX(cuenta) AS times FROM (SELECT * FROM (</pre>

	<pre> SELECT EXTRACT(YEAR FROM A.reg_date) AS year_p, EXTRACT(MONTH FROM A.reg_date) AS month_p, B.debtor AS dr_proteston, COUNT(B.debtor) AS cuenta FROM allegations A INNER JOIN tickets B ON A.obs_veh = B.obs1_veh AND A.obs_date = B.obs1_date AND A.tik_type = B.tik_type WHERE A.status = 'R' GROUP BY EXTRACT(YEAR FROM A.reg_date), EXTRACT(MONTH FROM A.reg_date), B.debtor ORDER BY year_p, month_p, cuenta)) GROUP BY year_p, month_p, dr_proteston ORDER BY year_p, month_p; </pre>
Funcionalidad	Consulta. Esta vista facilita los datos de los individuos con mas alegaciones rechazadas.
Pruebas	<p>Insertamos una allegation a la tabla como prueba y la view nos devuelve dicho resultado.</p> <pre> SQL> INSERT INTO observations VALUES ('16950EI', TO_TIMESTAMP('19-MAR-18 9.48.00.00.AM'),' ASC',200); 1 row created. SQL> INSERT INTO tickets VALUES ('16950EI', TO_TIMESTAMP('19-MAR-18 9.48.00.00.AM'), 'S' , TO_TIMESTAMP('17-MAR-18 9.48.00.00.AM'),NULL, NULL, 50,NULL,'R'); 1 row created. SQL> INSERT INTO ALLEGATIONS VALUES ('16950EI', TO_TIMESTAMP('19-MAR-18 9.48.00.00.AM'),' STAMP('17-MAR-18 9.48.00.00.AM'),'91988623P','R',TO_TIMESTAMP('17-MAR-18 9.48.00.00.AM')); 1 row created. SQL> SELECT * FROM PROTESTON; YEAR_P MONTH_P DR_PROTES TIMES ----- 2018 3 0 </pre> <p>Resultado esperado times: 0 ya que tickets no tiene como debtor el dni de la alegación porque no ha sido procesada.</p>

Vista 3	Tramos
Descripción	Tabla que registra cada tramo de carretera en el que la velocidad es inferior a la velocidad general de la vía (contiene la identificación de la vía, puntos de inicio y fin, y límite de velocidad en el tramo).
Algebra relacional	$\Pi_{A.road, \rho Km_point_Inicio(B.Km_point), \rho Km_point_Fin(A.Km_point), A.speedlim} \sigma_{A.road=B.road \wedge A.direction=B.direction \wedge ABS(B.Km_point-A.Km_point) \leq 5 \wedge ABS(B.km_point-A.Km_point) > 0} \neg A.road, B.Km_point$ <p>(radars A, radars B)</p>
SQL	<pre> CREATE OR REPLACE VIEW tramos AS SELECT DISTINCT A.road, B.Km_point AS Km_point_Inicio, A.Km_point AS Km_point_Fin, A.speedlim FROM radars A, radars B </pre>

Titulación: GRADO INGENIERIA INFORMATICA Año Académico: 2017/2018 Curso: 2º Asignatura: Ficheros y Bases de Datos Título: Memoria Práctica 2 – Consultas, vistas y disparadores	 uc3m Universidad Carlos III de Madrid
---	--

	WHERE A.road = B.road AND A.direction = B.direction AND ABS(B.Km_point-A.Km_point) <= 5 AND ABS(B.km_point- A.Km_point) >0 ORDER BY A.road, B.Km_point;																																								
Funcionalidad	Consulta. Esta vista facilita los datos de los tramos con radar.																																								
Pruebas	<pre>SELECT * FROM TRAMOS;</pre> <table><thead><tr><th>ROAD</th><th>KM_POINT_INICIO</th><th>KM_POINT_FIN</th><th>SPEEDLIM</th></tr></thead><tbody><tr><td>M50</td><td>16</td><td>18</td><td>100</td></tr><tr><td>M50</td><td>18</td><td>15</td><td>100</td></tr><tr><td>M50</td><td>18</td><td>16</td><td>100</td></tr><tr><td>M50</td><td>32</td><td>34</td><td>100</td></tr><tr><td>M50</td><td>34</td><td>32</td><td>100</td></tr><tr><td>M50</td><td>58</td><td>61</td><td>100</td></tr><tr><td>M50</td><td>61</td><td>58</td><td>100</td></tr><tr><td>M50</td><td>80</td><td>85</td><td>100</td></tr><tr><td>M50</td><td>85</td><td>80</td><td>100</td></tr></tbody></table> <pre>152 rows selected.</pre> <pre>SQL></pre>	ROAD	KM_POINT_INICIO	KM_POINT_FIN	SPEEDLIM	M50	16	18	100	M50	18	15	100	M50	18	16	100	M50	32	34	100	M50	34	32	100	M50	58	61	100	M50	61	58	100	M50	80	85	100	M50	85	80	100
ROAD	KM_POINT_INICIO	KM_POINT_FIN	SPEEDLIM																																						
M50	16	18	100																																						
M50	18	15	100																																						
M50	18	16	100																																						
M50	32	34	100																																						
M50	34	32	100																																						
M50	58	61	100																																						
M50	61	58	100																																						
M50	80	85	100																																						
M50	85	80	100																																						

Vista 4	Conductores_Avispados
Descripción	Los diez conductores cuyo promedio de velocidad se acerca más a los de la vía sin sobrepasarlo.
Algebra relacional	$\sigma_{\text{ROWNUM} \leq 10} \Pi_{\text{driver}(\text{vrd}), \text{promedio}(\text{AVG}(\text{percentage}))} \Join_{\text{vrd}(\text{vehicles.reg_driver}), \text{percentage}(100 * \text{observations.speed} / \text{radars.speedlim})} \sigma_{\text{speedlim} \leq \text{speedlim}} (\text{observations} \Join_{\text{observations.nPlate} = \text{vehicles.nPlate}} (\text{vehicles} \Join_{\text{observations.road} = \text{radars.road}} \text{AND observations.Km_point} = \text{radars.Km_point} \text{ AND observations.direction} = \text{radars.direction} (\text{radars})))$
Implementación SQL	<pre>CREATE OR REPLACE VIEW avispados AS SELECT * FROM (SELECT vrd AS driver, AVG(percentage) AS promedio FROM (SELECT vehicles.reg_driver AS vrd, 100*observations.speed/radars.speedlim AS percentage FROM (observations INNER JOIN vehicles ON observations.nPlate = vehicles.nPlate) INNER JOIN radars ON observations.road = radars.road AND observations.Km_point = radars.Km_point AND observations.direction = radars.direction) WHERE speed <= speedlim) GROUP BY vrd ORDER BY promedio) WHERE ROWNUM <=10;</pre>
Funcionalidad	Consulta. Esta vista facilita los datos de los conductores que cumplen la característica de avispados.

Titulación: GRADO INGENIERIA INFORMATICA

Año Académico: 2017/2018

Curso: 2º

Asignatura: Ficheros y Bases de Datos

Título: Memoria Práctica 2 – Consultas, vistas y disparadores



uc3m | Universidad Carlos III de Madrid

Pruebas

```
SELECT *FROM Avispados;
```

DRIVER	PROMEDIO
--------	----------

56193398M	88.4456522
-----------	------------

47591599E	89.6052632
-----------	------------

80771305C	89.69
-----------	-------

90553454Q	89.8148148
-----------	------------

54146779H	90.0625
-----------	---------

53018488S	90.1071429
-----------	------------

02626808R	90.4673913
-----------	------------

55533072D	90.4705882
-----------	------------

06729275G	90.49
-----------	-------

59015527B	90.5703125
-----------	------------

10 rows selected.

5. Disparadores

Trigger 1	Insertar_multa	
Descripción del diseño	Tabla asociada	observations
	Eventos	INSERT
	Condición	-
	Acción	(EN EL CODIGO SQL)
	Temporalidad	BEFORE
	Granularidad	EACH ROW
SQL	<pre> CREATE OR REPLACE TRIGGER insertar_multa FOR INSERT ON observations COMPOUND TRIGGER n NUMBER; row_changed observations%ROWTYPE; sp NUMBER(3); id VARCHAR2(9); vp NUMBER(3); obs2 OBS_TYPE; obs_anterior observations%ROWTYPE; radar_ant_vl NUMBER(3); BEFORE EACH ROW IS BEGIN row_changed.road := :NEW.road; row_changed.nPlate := :NEW.nPlate; row_changed.odatetime := :NEW.odatetime; row_changed.km_point := :NEW.km_point; row_changed.direction := :NEW.direction; row_changed.speed := :NEW.speed; -- multa por velocidad puntual -- speed supera a radar speedlim -- capturamos la velocidad lim del radar para (road, Km_point, y direction) END BEFORE EACH ROW; -- una vez insertada y para no entrar en tabla mutante AFTER STATEMENT IS --multa por velocidad maxima BEGIN SELECT speedlim INTO sp FROM radars WHERE row_changed.road = radars.road AND row changed.Km_point = radars.Km_point AND row_changed.direction = radars.direction; -- capturamos el DNI del owner del vehiculo SELECT owner INTO id FROM vehicles WHERE row_changed.nPlate = vehicles.nPlate; </pre>	

```

-- si la velocidad es superior a la
del radar, se crea una nueva fila en la tabla tickets
-- la cuantia se calcula con la
funcion amount_r, y se asigna al duenyo como deudor
IF row_changed.speed > sp THEN
-- la multa se inicia como
registrada, pero no hay fecha de envio (sent_date)
-- que no puede ser NULL, hay
que modificar la tabla
DBMS_OUTPUT.PUT_LINE(
'Generando multa por velocidad' );
INSERT INTO tickets
VALUES (row_changed.nPlate,
row_changed.odatetime, 'S', NULL, NULL, SYSDATE, NULL, NULL,
package_functions.amount_r(row_changed.speed, sp), id, 'R');
DBMS_OUTPUT.PUT_LINE( 'Multa
por velocidad insertada' );
END IF;

-- multa por velocidad en tramo
-- la velocidad calculada entre radares consecutivos que
distan menos de 5 km es mayor
-- es mayor que la velocidad limite de la carretera
-- capturamos la velocidad limite de la carretera que es la
que define multa en este caso
SELECT speed_limit INTO sp FROM roads WHERE
row_changed.road = name;
SELECT owner INTO id FROM vehicles WHERE
row_changed.nPlate = nPlate;
-- calculamos la velocidad promedio entre este radar y el
anterior usando la funcion o_bef_vehiculo
obs2 :=
package_functions.o_bef_vehiculo(row_changed.nPlate,
row_changed.odatetime);
SELECT * INTO obs_anterior FROM observations
WHERE nPlate = obs2.P_NPLATE AND odatetime =
obs2.P_ODATETIME;
IF (row_changed.road = obs_anterior.road AND
row_changed.direction = obs_anterior.direction) AND
(row_changed.km_point - obs_anterior.km_point) <= 5
THEN
vp := (row_changed.km_point -
obs_anterior.km_point)/((CAST(row_changed.odatetime AS DATE)
- CAST(obs_anterior.odatetime AS DATE))*24);
IF vp > sp THEN
DBMS_OUTPUT.PUT_LINE( 'Generando multa
por tramo' );
INSERT INTO tickets
VALUES (row_changed.nPlate,
row_changed.odatetime, 'T', row_changed.nPlate,
obs_anterior.odatetime,
SYSDATE, NULL,
NULL, package_functions.amount_t(obs_anterior.km_point,
obs_anterior.odatetime, row_changed.km_point,
row_changed.odatetime, sp), id, 'R');

```



```

                                DBMS_OUTPUT.PUT_LINE( 'Multa por tramo
insertada' );

                                END IF;

                                END IF;

-- multa por distancia minima
-- si el tiempo entre observaciones del mismo radar es menor
que 3.6 segundos
-- se multa al vehiculo que pertenece a la observacion
anterior
-- obetnemos observacion anterior del mismo radar
obs2 := package_functions.o_bef_radar(row_changed.odatetime,
row_changed.road, row_changed.km_point,
row_changed.direction);
SELECT * INTO obs_anterior FROM observations WHERE nPlate =
obs2.P_NPLATE AND odatetime = obs2.P_ODATETIME;
SELECT owner INTO id FROM vehicles WHERE vehicles.nPlate =
obs_anterior.nPlate;
IF (CAST(row_changed.odatetime AS DATE)-
CAST(obs_anterior.odatetime AS DATE))*24*60*60 < 3.6 THEN
DBMS_OUTPUT.PUT_LINE( 'Generando multa por distancia' );
INSERT INTO tickets
VALUES (obs_anterior.nPlate, obs_anterior.odatetime,'D',
row_changed.nPlate, row_changed.odatetime, SYSDATE, NULL,
NULL, package_functions.amount_d(row_changed.odatetime,
obs_anterior.odatetime), id, 'R');
DBMS_OUTPUT.PUT_LINE( 'Multa por distancia insertada'
);

                                END IF;

                                END AFTER STATEMENT;
END insertar_multa;

```

Pruebas

Output al insertar en la tabla observaciones con los 3 diferentes casos de multa posibles:

1.

```

SQL> -- TRIGGER SPEED
SQL> INSERT INTO observations VALUES ('16950EI', TO_TIMESTAMP('10-JAN-18 5.59.00.00.AM'),'A1',242, 'ASC',200);
Generando multa por velocidad
Multa por velocidad insertada

```

1 row created.

```
SQL> select * from tickets;
```

OBS1_VE	OBS1_DATE	TIK_TYPE	OBS2_VE	OBS2_DATE	SENT_DATE	PAY_DATE	P	AMOUNT	DEBTOR	S
16950EI	10-JAN-18 05.59.00.000000 AM	S			18-APR-18		1000	683529068	R	

2.

```

SQL> -- TRIGGER TRAMO
SQL> INSERT INTO observations VALUES ('16950EI', TO_TIMESTAMP('11-JAN-18 8.00.50.00.PM'),'M50',34, 'DES',20);
Generando multa por tramo
Multa por tramo insertada

```

1 row created.

```
SQL> select * from tickets;
```

OBS1_VE	OBS1_DATE	TIK_TYPE	OBS2_VE	OBS2_DATE	SENT_DATE	PAY_DATE	P	AMOUNT	DEBTOR	S
16950EI	10-JAN-18 05.59.00.000000 AM	S			18-APR-18		1000	683529068	R	
16950EI	11-JAN-18 08.00.50.000000 PM	T	16950EI	11-JAN-18 08.00.40.000000 PM	18-APR-18		6000	683529068	R	

3.

SQL> -- TRIGGER DISTANCIA

SQL> INSERT INTO observations VALUES ('64930EA', TO_TIMESTAMP('11-JAN-18 8.00.52.00.PM'),'M50',34, 'DES',20);

Generando multa por distancia

Multa por distancia insertada

1 row created.

SQL> select * from tickets;

OBS1_VE	OBS1_DATE	TIK_TYPE	OBS2_VE	OBS2_DATE	SENT_DATE	PAY_DATE	P	AMOUNT	DEBTOR	S
16950EI	10-JAN-18 05.59.00.000000 AM	S			18-APR-18		1000	683529068	R	
16950EI	11-JAN-18 08.00.50.000000 PM	T	16950EI	11-JAN-18 08.00.40.000000 PM	18-APR-18		6000	683529068	R	
16950EI	11-JAN-18 08.00.50.000000 PM	D	64930EA	11-JAN-18 08.00.52.000000 PM	18-APR-18		160	683529068	R	

Trigger 2	Procesar alegación	
Descripción del diseño	Tabla asociada	allegations
	Eventos	INSERT
	Condición	-
	Acción	(EN EL CODIGO SQL)
	Temporalidad	(Compound Trigger) BEFORE AFTER
	Granularidad	EACH ROW
SQL	<pre>CREATE OR REPLACE TRIGGER allegation FOR INSERT ON allegations COMPOUND TRIGGER n NUMBER; row_changed allegations%ROWTYPE; -- before inserta una nueva alegacion que puede ser R o A BEFORE EACH ROW IS -- aqui no podemos acceder o modificar la tabla alegaciones BEGIN -- almacenamos datos nueva fila row_changed.obs_veh := :NEW.obs_veh; row_changed.obs_date := :NEW.obs_date; row_changed.tik_type := :NEW.tik_type; row_changed.new_debtor := :NEW.new_debtor; row_changed.reg_date := SYSDATE; -- averiguamos si el nuevo deudor es un driver asignado SELECT COUNT(driver) INTO n FROM assignments WHERE nPlate = :NEW.obs_veh AND driver = :NEW.new_debtor; -- si el nuevo deudor no es un conductor asignado, se rechaza IF n = 0 THEN :NEW.status := 'R'; --cambiamos el default status que es 'U' :NEW.exec_date := SYSDATE; DBMS_OUTPUT.PUT_LINE('Rechazada: no es un conductor asignado'); ELSE</pre>	

```
-- si es nuevo deudor es un conductor asignado, se aprueba inicialmente
:NEW.status := 'A';
:NEW.exec_date := SYSDATE;
DBMS_OUTPUT.PUT_LINE( 'Aprobada' );
-- con las alegaciones Aprobadas hay que cambiar el estado del ticket
-- a registrado y actualizar el deudor
UPDATE tickets
SET debtor = :NEW.new_debtor, state = 'R'
WHERE obs1_veh = :NEW.obs_veh AND obs1_date = :NEW.obs_date
AND tik_type = :NEW.tik_type;
END IF;
END BEFORE EACH ROW;
-- una vez insertada, buscamos si la misma multa ha sido allegada previamente
AFTER STATEMENT IS
-- aqui si podemos acceder o modificar la tabla alegaciones
-- si hay mas de una alegacion para el mismo ticket, cambiamos la alegacion a 'U'
BEGIN
SELECT COUNT(reg_date) INTO n FROM allegations WHERE obs_veh = row_changed.obs_veh AND obs_date = row_changed.obs_date AND tik_type = row_changed.tik_type AND new_debtor = row_changed.new_debtor;
IF n > 1 THEN
DBMS_OUTPUT.PUT_LINE( 'En estudio por alegacion recurrente' );
UPDATE allegations
SET status = 'U', exec_date = NULL
WHERE obs_veh = row_changed.obs_veh AND obs_date = row_changed.obs_date
AND tik_type = row_changed.tik_type AND reg_date = row_changed.reg_date;
END IF;
END AFTER STATEMENT;
END allegation;
```

Titulación: GRADO INGENIERIA INFORMATICA

Año Académico: 2017/2018

Curso: 2º

Asignatura: Ficheros y Bases de Datos

Título: Memoria Práctica 2 – Consultas, vistas y disparadores




uc3m | Universidad Carlos III de Madrid

Pruebas	<pre>SQL> SQL> INSERT INTO allegations VALUES('64930EA', TO_TIMESTAMP('16-APR-18 8.33.00.00.PM'),'S', SYSDATE, '69049869M','U',NULL); Rechazada: no es un conductor asignado 1 row created. SQL> INSERT INTO allegations VALUES('64930EA', TO_TIMESTAMP('16-APR-18 8.33.00.00.PM'),'S', SYSDATE, '29480068W','U',NULL); Aprobada 1 row created. SQL> INSERT INTO allegations VALUES('64930EA', TO_TIMESTAMP('16-APR-18 8.33.00.00.PM'),'S', SYSDATE, '65572592Y','U',NULL); Aprobada 1 row created. SQL> INSERT INTO allegations VALUES('64930EA', TO_TIMESTAMP('16-APR-18 8.33.00.00.PM'),'S', SYSDATE, '29480068W','U',NULL); Aprobada En estudio por alegacion recurrente 1 row created. SQL> SELECT * FROM allegations; OBS_VEH OBS_DATE TIK_TYPE REG_DATE NEW_DEBTO S EXEC_DATE ----- 64930EA 16-APR-18 08.33.00.000000 PM S 17-APR-18 69049869M R 17-APR-18 64930EA 16-APR-18 08.33.00.000000 PM S 17-APR-18 29480068W A 17-APR-18 64930EA 16-APR-18 08.33.00.000000 PM S 17-APR-18 65572592Y A 17-APR-18 64930EA 16-APR-18 08.33.00.000000 PM S 17-APR-18 29480068W U SQL> SELECT * FROM TICKETS WHERE OBS1_VEH = '64930EA'; OBS1_VE OBS1_DATE TIK_TYPE OBS2_VE OBS2_DATE SENT_DATE PAY_DATE P AMOUNT DEBTOR S ----- 64930EA 16-APR-18 08.33.00.000000 PM S 1000 29480068W R</pre> <p>Se realiza una inserción en la tabla allegations para disparar el trigger y se obtiene el resultado esperado tras procesarla.</p>
---------	---


Trigger 3	A rey muerto	
Descripción del diseño	Tabla asociada	vehicles
	Eventos	UPDATE
	Condición	-
	Acción	EN EL CODIGO SQL
	Temporalidad	BEFORE
	Granularidad	EACH ROW
SQL	<pre>CREATE OR REPLACE TRIGGER rey muerto BEFORE UPDATE OF reg_driver ON vehicles FOR EACH ROW DECLARE n NUMBER; BEGIN IF (:NEW.reg_driver IS NULL) THEN -- asignamos como nuevo conductor el de mas antigüedad de la tabla asignados -- para este vehiculo SELECT COUNT(driver) INTO n FROM assignments WHERE nPlate = :NEW.nPlate; IF n > 0 THEN SELECT driver INTO :NEW.reg_driver FROM (SELECT assignments.driver, drivers.lic_date FROM assignments INNER JOIN drivers ON assignments.driver = drivers.DNI WHERE assignments.nPlate = :NEW.nPlate</pre>	

	<pre> ORDER BY lic_date DESC) WHERE ROWNUM = 1; ELSE RAISE_APPLICATION_ERROR(- 20110, 'No hay conductor asignado'); END IF; END IF; END;</pre>
Pruebas	<p>Se elimina el conductor asociado poniéndolo a NULL entonces se dispara el trigger y actualiza la información al conductor asociado con licencia mas antigua.</p> <pre> SQL> select * from vehicles where nplate = '64930EA'; NPLATE VIN MAKE MODEL COLOR ----- REG_DATE MOT_DATE REG_DRIVE OWNER ----- 64930EA IQQ64123Q58925888 Merche Berlin negro 14-APR-03 30-AUG-11 65572592Y 99181271G SQL> UPDATE VEHICLES SET REG_driver = NULL WHERE NPLATE = '64930EA'; 1 row updated. SQL> select * from vehicles where nplate = '64930EA' 2 ; NPLATE VIN MAKE MODEL COLOR ----- REG_DATE MOT_DATE REG_DRIVE OWNER ----- 64930EA IQQ64123Q58925888 Merche Berlin negro 14-APR-03 30-AUG-11 29480068W 99181271G Como se puede observar se ha asignado dentro de las posibilidades el que tiene licencia mas antigua. SQL> select * from assignments; DRIVER NPLATE ----- 29480068W 64930EA 65572592Y 64930EA 68352906B 16950EI 99181271G 64930EA SQL> select * from drivers where dni = '29480068W' OR dni = '65572592Y' or dni = '99181271G' DNI LIC_DATE LIC ----- 29480068W 29-OCT-11 C1 65572592Y 21-APR-05 B 99181271G 02-MAY-96 D Error en caso de que no exista asignacion para tal vehiculo: SQL> update vehicles set reg_driver =NULL WHERE NPLATE = '9959U0U'; update vehicles set reg_driver =NULL WHERE NPLATE = '9959U0U' * ERROR at line 1: ORA-20110: No hay conductor asignado ORA-06512: at "FSDB175.REY_MUERTO", line 19 ORA-04088: error during execution of trigger 'FSDB175.REY_MUERTO'</pre>


Trigger 4	Restricciones	
Descripción del diseño	Tabla asociada	radars
	Eventos	INSERT OR UPDATE
	Condición	-

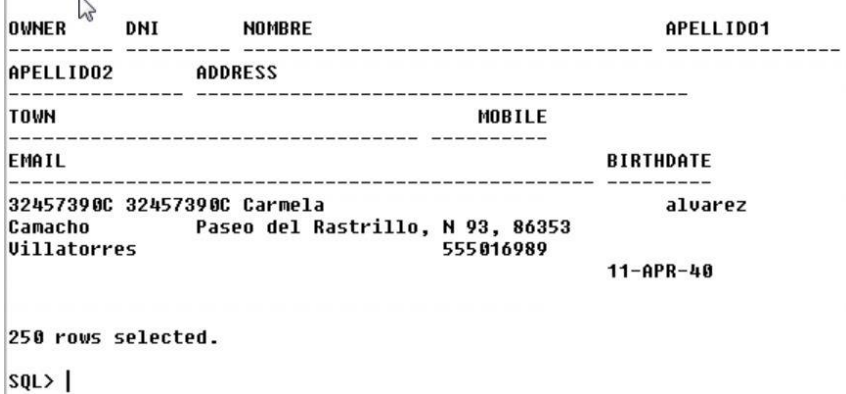
Titulación: GRADO INGENIERIA INFORMATICA Año Académico: 2017/2018 Curso: 2º Asignatura: Ficheros y Bases de Datos Título: Memoria Práctica 2 – Consultas, vistas y disparadores	 uc3m Universidad Carlos III de Madrid
---	--


	Acción Temporalidad Granularidad	EN CODIGO SQL BEFORE EACH ROW
SQL	<pre> CREATE OR REPLACE TRIGGER RADAR_SPEED BEFORE INSERT OR UPDATE ON radars FOR EACH ROW DECLARE sp NUMBER(3,0); BEGIN SELECT speed_limit INTO sp FROM roads WHERE name = :NEW.road; IF :NEW.speedlim > sp THEN RAISE_APPLICATION_ERROR(-20111,'Velocidad radar no puede ser mayor que velocidad road'); END IF; END; -- conductores al menos 18 anyos CREATE OR REPLACE TRIGGER MENOR_EDAD BEFORE INSERT OR UPDATE ON drivers FOR EACH ROW DECLARE bd DATE; BEGIN SELECT birthdate INTO bd FROM persons WHERE DNI = :NEW.DNI; IF FLOOR((SYSDATE - bd)/365) < 18 THEN RAISE_APPLICATION_ERROR(-20112, 'Edad no puede ser menor que 18'); END IF; END; </pre>	
Pruebas	<p>Excepcion 1 :</p> <p>Se intenta insertar una velocidad de radar de 200 mientras que la de la carretera es menor. Salta el trigger como era esperado.</p> <pre> SQL> INSERT INTO RADARS VALUES ('A2', '32', 'ASC',200); INSERT INTO RADARS VALUES ('A2', '32', 'ASC',200) * ERROR at line 1: ORA-20111: Velocidad radar no puede ser mayor que velocidad road ORA-06512: at "FSDB175.RADAR_SPEED", line 5 ORA-04088: error during execution of trigger 'FSDB175.RADAR_SPEED' </pre> <p>Excepcion 2:</p> <p>Se trata de insertar un conductor que tiene como fecha de nacimiento el year 2000. El resultado es el esperado disparandose el trigger.</p> <pre> SQL> INSERT INTO DRIVERS VALUES ('77777777P', '11-SEP-00', 'B'); INSERT INTO DRIVERS VALUES ('77777777P', '11-SEP-00', 'B') * ERROR at line 1: ORA-20112: Edad no puede ser menor que 18 ORA-06512: at "FSDB175.MENOR_EDAD", line 6 ORA-04088: error during execution of trigger 'FSDB175.MENOR_EDAD' </pre>	

Titulación: GRADO INGENIERIA INFORMATICA Año Académico: 2017/2018 Curso: 2º Asignatura: Ficheros y Bases de Datos Título: Memoria Práctica 2 – Consultas, vistas y disparadores	 uc3m Universidad Carlos III de Madrid
---	--

6. Diseño externo (Opcional)

Perfil Rel_Publicas		
Rol	<pre>CREATE PROFILE rel_publicas LIMIT SESSIONS_PER_USER UNLIMITED; CREATE USER rp_name IDENTIFIED BY rp_password PROFILE rel_publicas; GRANT SELECT ON vehicles TO rp_name; GRANT SELECT ON rp_conductores_view TO rp_name; GRANT SELECT ON rp_owners_view TO rp_name; GRANT SELECT ON rp_asignaciones_view TO rp_name;</pre>	
Vista conductores	Algebra Relacional	Π drivers.DNI, drivers.lic_date, drivers.lic_type, pnombre (persons.name), ppersons.surn_1(apellido1), p persons.surn_2 (apellido2), persons.address, persons.town, persons.mobile, persons.email, persons.birthdate \neg apellido1, apellido2, nombre (drivers $\theta_{\text{drivers.DNI=persons.DNI}}$ persons)
	SQL	<pre>CREATE OR REPLACE VIEW rp_conductores_view AS SELECT drivers.DNI, drivers.lic_date, drivers.lic_type, persons.name AS nombre, persons.surn_1 AS apellido1, persons.surn_2 AS apellido2, persons.address, persons.town, persons.mobile, persons.email, persons.birthdate FROM drivers INNER JOIN persons ON drivers.DNI = persons.DNI ORDER BY apellido1, apellido2, nombre;</pre>
	Prueba	
Vista dueños	Algebra Relacional	Π vehicles.owner, persons.DNI, pnombre (persons.name), ppersons.surn_1(apellido1), p persons.surn_2 (apellido2), persons.address, persons.town, persons.mobile, persons.email, persons.birthdate \neg apellido1, apellido2, nombre (vehicles $\theta_{\text{vehicles.owner=persons.DNI}}$ persons)
	SQL	<pre>CREATE OR REPLACE VIEW rp_owners_view AS SELECT vehicles.owner, persons.DNI, persons.name AS nombre, persons.surn_1 AS apellido1, persons.surn_2</pre>

		<p>AS apellido2, persons.address, persons.town, persons.mobile, persons.email, persons.birthdate FROM vehicles INNER JOIN persons ON vehicles.owner = persons.DNI ORDER BY apellido1, apellido2, nombre;</p>
	Prueba	
Vista asignaciones	Algebra Relacional	$\Pi_{dr_DNI, dr_asgvehiculo, cond_habitual} \neg dr_DNI, dr_asgvehiculo$ $((\Pi_{pdr_DNI(driver_DNI), pdr_asgvehiculo(veh_plate), pcond_habitual('NO')} (\Pi_{pdriver_DNI(assignments.driver), pveh_placel(assignments.nPlate)} \sigma_{assignments.driver \neq vehicles.reg_driver} (assignments \theta_{assignments.nPlate=vehicles.nPlate} vehicles) \cup (\Pi_{pdr_DNI(driver_DNI), pdr_asgvehiculo(veh_plate), pcond_habitual('SI')} (\Pi_{pdriver_DNI(assignments.driver), pveh_placel(assignments.nPlate)} \sigma_{assignments.driver=vehicles.reg_driver} (assignments \theta_{assignments.nPlate=vehicles.nPlate} vehicles))))))$
	SQL	<pre>CREATE OR REPLACE VIEW rp_asignaciones_view AS SELECT dr_DNI, dr_asgvehiculo, cond_habitual FROM(-- lista de drivers que no son regular driver SELECT driver_DNI as dr_DNI, veh_plate AS dr_asgvehiculo, 'NO' AS cond_habitual FROM (SELECT assignments.driver as driver_DNI, assignments.nPlate as veh_plate FROM assignments INNER JOIN vehicles ON assignments.nPlate = vehicles.nPlate WHERE assignments.driver != vehicles.reg_driver) UNION -- lista de drivers que si son regular driver SELECT driver_DNI as dr_DNI, veh_plate AS dr_asgvehiculo, 'SI' AS cond_habitual FROM (</pre>

Titulación: GRADO INGENIERIA INFORMATICA Año Académico: 2017/2018 Curso: 2º Asignatura: Ficheros y Bases de Datos Título: Memoria Práctica 2 – Consultas, vistas y disparadores	 uc3m Universidad Carlos III de Madrid
---	--

		<pre>SELECT assignments.driver as driver_DNI, assignments.nPlate as veh_plate FROM assignments INNER JOIN vehicles ON assignments.nPlate = vehicles.nPlate WHERE assignments.driver = vehicles.reg_driver)) ORDER BY dr_DNI, dr_asgvehiculo;</pre>
	Prueba	<pre>SQL> insert into assignments values ('68352906B','16950EI'); 1 row created. SQL> select * from rp_asignaciones_view; DR_DNI DR_ASGU CO ----- 68352906B 16950EI SI</pre> <p>Tras insertar en la tabla asignaciones se muestra la view esperada.</p>
Vista buenagente	Algebra Relacional	$\Pi \text{ DNI } \sigma_{\text{NOT EXISTS}(\sigma_{\text{allegations.new_debtor=persons.DNI AND allegations.status='R'}})(\text{persons})}$
	SQL	<pre>CREATE OR REPLACE VIEW buenagente_view AS --personas que no aparecen en la tabla alegaciones (si alegacion es Rechazada) SELECT DNI FROM persons WHERE NOT EXISTS (SELECT * FROM allegations WHERE allegations.new_debtor = persons.DNI AND allegations.status = 'R');</pre>
	Prueba	<pre>SELECT * FROM BUENAGENTE_VIEW; SQL> INSERT INTO ALLEGATIONS VALUES ('16950EI', TO_TIMESTAMP('19-MAR-18 9.48.00.AM'),' STAMP('17-MAR-18 9.48.00.AM'),'91988623P','R',TO_TIMESTAMP('17-MAR-18 9.48.00.AM'))); 1 row created. Tras la inserccion en alegaciones un elemento de la tabla personas esta en la tabla alegaciones por lo que de las 248 personas totales en los datos el resultado de 247 concuerda con lo esperado. DNI ----- 88062522F 62526264M 08914907S 35235590G 23939454L 247 rows selected.</pre>

Perfil Administrativo	
Rol	<pre>CREATE PROFILE administrativo LIMIT SESSIONS_PER_USER UNLIMITED; CREATE USER ad_name IDENTIFIED BY ad_password PROFILE administrativo; GRANT SELECT ON ad_sanc_impagadas_view TO ad_name; GRANT SELECT ON ad_notificacion_view TO ad_name; GRANT SELECT ON ad_ult_infraccion view TO ad name;</pre>

Titulación: GRADO INGENIERIA INFORMATICA

Año Académico: 2017/2018

Curso: 2º

Asignatura: Ficheros y Bases de Datos

Título: Memoria Práctica 2 – Consultas, vistas y disparadores



uc3m | Universidad Carlos III de Madrid

	GRANT SELECT ON allegations TO ad_name;																																			
Vista sanciones impagadas	Algebra Relacional	Π amount, amount*2, ppenalizacion (amount*2), ptotal_multa (amount*3) σ state='N'(tickets)																																		
	SQL	CREATE OR REPLACE VIEW ad_sanc_impagadas_view AS SELECT amount, amount*2 AS penalizacion, amount*3 AS total_multa FROM tickets WHERE state = 'N';																																		
	Prueba	Se inserta en la tabla de multas un ticket que no ha sido pagado y el resultado de la view lo muestra. SQL> INSERT INTO observations VALUES ('16950EI', TO_TIMESTAMP('20-MAR-18 10.48.00.00.AM'),'A1',242, 'ASC',200); 1 row created. SQL> INSERT INTO tickets VALUES ('16950EI', TO_TIMESTAMP('20-MAR-18 10.48.00.00.AM'), 'S', NULL,NULL, TO_TIMESTAMP('17-MAR-18 9.48.00.00.AM'),NULL, NULL, 50,NULL,'N'); 1 row created. SELECT * FROM AD_SANC_IMPAGADAS_VIEW; <table><thead><tr><th>AMOUNT</th><th>PENALIZACION</th><th>TOTAL_MULTA</th></tr></thead><tbody><tr><td>50</td><td>100</td><td>150</td></tr></tbody></table>	AMOUNT	PENALIZACION	TOTAL_MULTA	50	100	150																												
AMOUNT	PENALIZACION	TOTAL_MULTA																																		
50	100	150																																		
Vista notificación	Algebra Relacional	Π nPlate, owner_DNI, sent_date pnombre (persons.name), ppersons.surn_1(apellido1), ρ persons.surn_2 (apellido2), email, mobile \neg email, mobile, address (Π nPlate, powner_DNI (owner), sent_date σ state='N' (tickets $\theta_{\text{obs1_veh}=n\text{Plate}}$ vehicles) $\theta_{\text{owner_DNI}=persons.DNI}$ (persons)))																																		
	SQL	CREATE OR REPLACE VIEW ad_notificacion_view AS SELECT nPlate, owner_DNI, sent_date, name AS owner_name, surn_1 AS owner_appellido1, surn_2 AS owner_appellido2, email, mobile, address FROM (SELECT nPlate, owner AS owner_DNI, sent_date FROM tickets INNER JOIN vehicles ON obs1_veh = nPlate WHERE state = 'N') INNER JOIN persons ON owner_DNI = persons.DNI ORDER BY email, mobile, address;																																		
	Prueba	SELECT * FROM AD_NOTIFICACION_VIEW; <table><thead><tr><th>NPLATE</th><th>OWNER_DNI</th><th>SENT_DATE</th><th>OWNER_NAME</th><th>OWNER_APELLIDO</th></tr></thead><tbody><tr><td>16950EI 68352906B</td><td>17-MAR-18</td><td>Maria de la Prudencia</td><td>Amancio</td><td>555895428</td></tr><tr><td colspan="5">Faucet</td></tr><tr><td colspan="5">Paseo de los Tulipanes, N 97, 28617</td></tr><tr><td>16950EI 68352906B</td><td>17-MAR-18</td><td>Maria de la Prudencia</td><td>Amancio</td><td>555895428</td></tr><tr><td colspan="5">Faucet</td></tr><tr><td colspan="5">Paseo de los Tulipanes, N 97, 28617</td></tr></tbody></table> Se muestra el resultado procedente del ticket insertado en las pruebas anteriores.	NPLATE	OWNER_DNI	SENT_DATE	OWNER_NAME	OWNER_APELLIDO	16950EI 68352906B	17-MAR-18	Maria de la Prudencia	Amancio	555895428	Faucet					Paseo de los Tulipanes, N 97, 28617					16950EI 68352906B	17-MAR-18	Maria de la Prudencia	Amancio	555895428	Faucet					Paseo de los Tulipanes, N 97, 28617			
NPLATE	OWNER_DNI	SENT_DATE	OWNER_NAME	OWNER_APELLIDO																																
16950EI 68352906B	17-MAR-18	Maria de la Prudencia	Amancio	555895428																																
Faucet																																				
Paseo de los Tulipanes, N 97, 28617																																				
16950EI 68352906B	17-MAR-18	Maria de la Prudencia	Amancio	555895428																																
Faucet																																				
Paseo de los Tulipanes, N 97, 28617																																				
Vista ult_infraccion	Algebra Relacional	Π ρ matricula(obs1_veh), ρ fecha(obs1_date) σ rownum=1($\Pi_{\neg \text{obs1_date}}$ (tickets))																																		

Titulación: GRADO INGENIERIA INFORMATICA

Año Académico: 2017/2018

Curso: 2º

Asignatura: Ficheros y Bases de Datos

Título: Memoria Práctica 2 – Consultas, vistas y disparadores



uc3m | Universidad Carlos III de Madrid

	SQL	<pre>CREATE OR REPLACE VIEW ad_ult_infraccion_view AS SELECT obs1_veh as matricula, obs1_date as fecha FROM(SELECT * FROM tickets ORDER BY obs1_date) WHERE ROWNUM =1;</pre>
	Prueba	<p>SELECT * FROM AD_ULT_INFRACCION_VIEW; Se muestra el resultado procedente del ticket insertado en las pruebas anteriores.</p> <pre>MATRICU ----- FECHA ----- 16950EI 18-MAR-17 09.48.00.000000 AM</pre>

7. Procedimiento

Procedimiento	envia_sanciones
Descripción	Generar sanciones cometidas por día
Funcionamiento	Este procedimiento toma las multas del día anterior en estado "Registrada", muestra un mensaje en pantalla, y cambia el estado a "Enviada"
SQL	<pre> CREATE OR REPLACE PROCEDURE envia_sanciones IS -- Declaración de variables locales CURSOR c1 IS SELECT * FROM tickets WHERE state = 'R' AND TRUNC(obs1_date, 'day') = TRUNC(SYSTIMESTAMP-1, 'day'); BEGIN -- cursor que recorre cada fila de la tabla para aquellas multas -- en estado 'R' y cuya fecha es del día anterior al actual -- For Loop para mostrar en pantalla las multas del día anterior FOR rec in c1 LOOP -- actualiza las columnas: state y sent_date rec.state := 'E'; rec.sent_date := SYSDATE; DBMS_OUTPUT.PUT_LINE ('Vehiculo: ' rec.obs1_veh '; Fecha: ' rec.obs1_date 'Tipo multa: ' rec.tik_type); DBMS_OUTPUT.PUT_LINE ('Conductor: ' rec.debtor); DBMS_OUTPUT.PUT_LINE ('Cuantia multa: ' rec.amount '; Fecha limite pago: ' SYSDATE + 20); DBMS_OUTPUT.PUT_LINE ('----- -----'); END LOOP; END;</pre>
Prueba	<p>Inicialmente permitimos que se envíen por pantalla avisos y se envía la sanción correspondiente a el ticket usado para los ejemplos.</p> <pre> SQL> SET SERVER OUTPUT ON; SP2-0158: unknown SET option "SERVER" SQL> SET SERVEROUTPUT ON; SQL> EXEC EN VIA_SANCIONES; Vehiculo: 16950EI; Fecha: 17-APR-18 09.48.00.000000 AM Tipo multa: S Conductor: Cuantia multa: 60; Fecha limite pago: 08-MAY-18 ----- Vehiculo: 16950EI; Fecha: 18-APR-18 09.48.00.000000 AM Tipo multa: S Conductor: Cuantia multa: 60; Fecha limite pago: 08-MAY-18 ----- PL/SQL procedure successfully completed.</pre>

8. Conclusiones

En este proyecto hemos realizado todos los diferentes apartados propuestos en el enunciado consiguiendo un código capaz de manejar la base de datos propuesta de manera eficaz y eficiente.

Nuestro diseño se diferencia por la gran cobertura que hemos realizado para las diferentes subcategorías, hemos llevado a cabo una batería exhaustiva de pruebas teniendo en cuenta posibles casos como divisiones no permitidas o números negativos en el desarrollo de algunas funciones, por ejemplo.

Todos los diferentes apartados han sido comprobados cerciorándonos de que el código presentado funcione correctamente y tenga resultados coherentes. Para ello hemos utilizado estrategias como cambiar la fecha del propio sistema o ayudarnos de otros apartados para conseguir datos, como el disparador que inserta multas o el procedimiento que genera sanciones y utilizarlos como pruebas.

Durante el proceso nos hemos apoyado principalmente en recursos en la red como <https://docs.oracle.com/> o <https://www.w3schools.com/sql/default.asp> que proporcionan una variedad amplia de ejemplos y documentación útil.

Siendo cierto que creemos haber aprendido el manejo de sql y la sintaxis gracias al trabajo realizado en las prácticas, si creemos que el proceso podría haber sido más ameno si se le proporcionara al estudiante una guía simple (material de apoyo) sobre algunas cuestiones concretas de Oracle/sql como tipos de fecha o diferencia de formatos. Sería útil ya que es algo que quita cierto tiempo al tener que buscar información en la red e ir un poco ciegas. También hacemos hincapié en que nos hubiera facilitado un enunciado algo más extenso o con más detalles sobre que quiere el usuario para aun tomando decisiones de diseño que no hubiese demasiadas dudas en cuanto al esquema del problema.

Por otro lado, creemos que hemos aprendido a manejar bien la herramienta y a saber buscar información sobre la misma para poder solventar cualquier problema que se presente. En cuanto al tamaño de los problemas o el tiempo no han sido un inconveniente ya que se ha proporcionado suficiente margen aun así hemos de comentar que es cierto que hemos tenido varias veces problemas para acceder al programa sqlplus vía Aula Virtual siendo esto un inconveniente.