EXAMEN HÉCTOR GARCÍA GONZÁLEZ

1)

USE exam

GO

ALTER TABLE hotel

ADD CONSTRAINT FK\_CATEGORIA CHECK (categoria LIKE '[\*][\*\*][\*\*\*][\*\*\*\*][\*\*\*\*\*]')

GO

2)

USE exam;

GO

CREATE VIEW addCliente2 AS SELECT cliente.codcli, cliente.nombre ,cliente.apellido, cliente.dni, cliente.ciucli

FROM cliente

WHERE cliente.ciucli = 'Logroño'

GO

USE exam;

GO

INSERT INTO addCliente2

VALUES (7, 'hector','garcia','11221122j','Madrid')

GO

SI, si usamos WITH CHECK OPTION

USE exam;

GO

CREATE VIEW addCliente1 AS SELECT cliente.codcli, cliente.nombre ,cliente.apellido, cliente.dni

FROM cliente

WHERE cliente.ciucli = 'Logroño' WITH CHECK OPTION

GO

Ahora solo admitirá clientes que sean de logroño

3)

USE exam

GO

CREATE USER pepe WITHOUT LOGIN

GO ---CREAR PERMISOS SOBRE VISTA addCliente2

GRANT SELECT, INSERT ON addCliente2 TO pepe

go

--con with grant option sginifica que se asignan todos los permisos

--al usuario (select, insert, delete, etc)

4)

USE exam;

GO

CREATE UNIQUE INDEX Indice\_DNI\_Clientes ON Cliente(dni)

GO

Con un índice creamos como una segunda tabla auxiliar en la que solo tendremos la información que nosotros queramos para facilitarnos las consultas. Con el UNIQUE nos aseguramos que el DNI del cliente no se repita en la consulta y así evitar duplicaciones.

5) USE exam;

GO

ALTER TABLE Hotel

ADD ANTIGUEDAD\_HOTEL AS DATEDIFF(YY,GETDATE(),fechainaug)

GO

6)

USE exam

ALTER TABLE estancia

ADD CONSTRAINT ch\_fechaentrada CHECK (fechaentrada > fechasalida)

GO

7)

USE exam;

SELECT c.nombre, c.apellido ,TIPO =

CASE

WHEN h.fechainaug > '01/01/1990' THEN 'Nuevo'

WHEN h.fechainaug >'01/01/1980' AND h.fechainaug < '31/12/1989' THEN 'Moderno'

WHEN h.fechainaug < '01/01/1980' THEN 'Clasico'

ELSE 'no calificacion'

END

FROM cliente c, hotel h, estancia e

WHERE c.codcli IN (SELECT e.codcli FROM estancia WHERE e.codhot IN (select codhot from hotel))

8)

USE exam

GO

UPDATE hotel

SET categoria = '\*\*\*'

WHERE hotel.codhot IN (SELECT codhot FROM estancia

WHERE estancia.codhot IN (SELECT codhot FROM habitacion

WHERE habitacion.tipohab = 'suite' AND habitacion.precionoche > 200) )

;

9)

USE exam;

GO

CREATE VIEW PTO8 AS SELECT c.nombre, e.numhab, h.nomhot

FROM cliente c, estancia e, hotel h

WHERE c.codcli = e.codcli AND e.codhot = h.codhot

GO

10)

USE exam

GO

CREATE TABLE empleados

(

id\_empleado INT IDENTITY (3,1) PRIMARY KEY,

nombempleado VARCHAR (50),

apellempleado VARCHAR (50),

codigohotel SMALLINT NOT NULL

)

GO

ALTER TABLE estancia

ADD cod\_empleado INT

GO

ALTER TABLE estancia

ADD CONSTRAINT fk\_cod\_empleado FOREIGN KEY REFERENCES empleados(id\_empleado)

ON DELETE NO ACTION

GO

ALTER TABLE empleados

add constraint fk\_cod\_hotel FOREIGN KEY (codhot)

REFERENCES hotel(codhot) ON DELETE NO ACTION

GO

INSERT INTO empleados SELECT 'Juan','Perez', codhot

FROM hotel WHERE hotel.nomhot= 'Ritz'

GO

11) Una transacción en SQL es un conjunto de operaciones que se realizan o efectúan en bloque (Ejemplo : realizar 3 insert a la vez o 3 Update). En la transacción si todas las operaciones se realizan correctamente, los datos se guardaran correctamente ,pero si una de las operaciones falla, dara error y entonces el resto de operaciones no se efectuara. Ejemplo: 3 transferencias bancarias efectuadas a la vez, si una falla, el resto no se ejecutara a no ser que se programe para que así sea.

Propiedades:

-Atomicidad: Las operaciones se realizan todas en conjunto, no por separado.

-Consistencia: Los datos deben permanecer integros.

-Aislamiento: Los cambios realizados durante las operaciones no afectan al resto de datos no relacionados con la transacción.

-Durabilidad: Una vez efectuadas las operaciones, estos cambios perduran y son permanentes en el tiempo.