



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CIENCIAS

Reporte Proyecto

ALUMNOS

Morales Hidalgo Pedro - 319267334

PROFESOR

Víctor Manuel Corza Vargas

AYUDANTES

Diana Irma Canchola Hernández

Oscar José Hernández Sánchez

Gibrán Aguilar Zuñiga

ASIGNATURA

Fundamentos de bases de datos

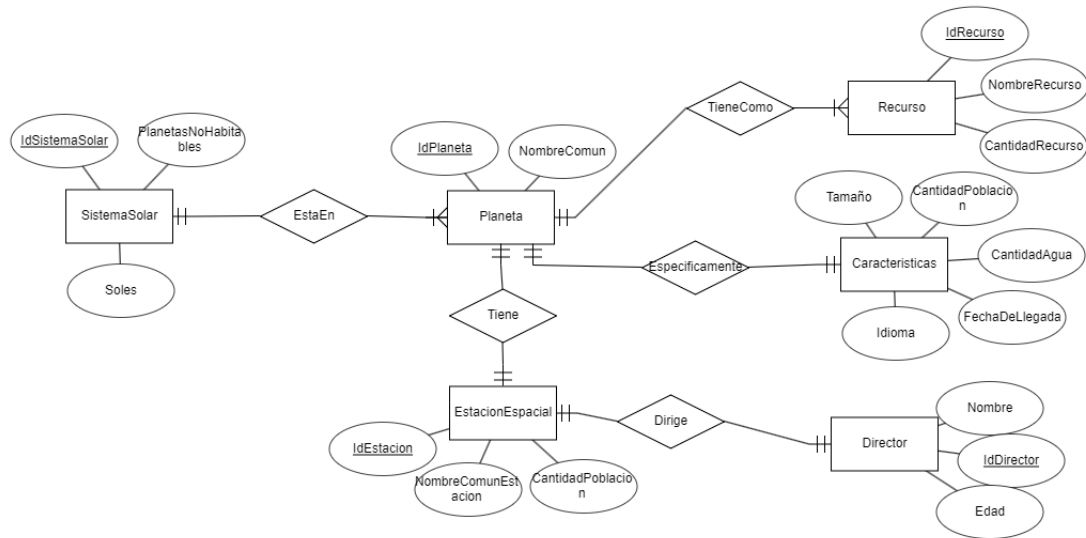
11 de diciembre de 2023

1. Lista de requerimientos

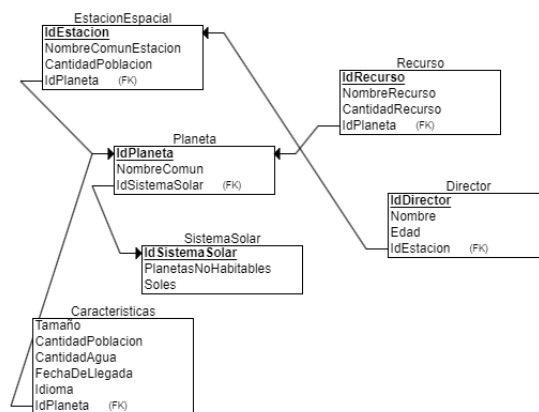
Para la base de datos pensé en que puede que mañana hayamos descubierto la forma de viajar a los planetas y sistemas solares sin tener que preocuparnos por el tiempo, entonces necesitamos saber cuales sistemas han sido visitados y de estos que planetas han sido habitados, juntos con una serie de atributos, como los recursos que hay en este, cantidad de población que hay, cantidad de agua, etc. Mi ser mas profundo cree en el warhammer40k pido perdon. Bueno yo desarrolle e hice las pruebas en postgres 16, lo ejecute en windows 10, y vaya que si son muchos insert. Ya lo puse en la parte de las aclaraciones, pero sin duda me arrepenti de haber metido sistemas solares, pero creo que le da un poco de magia a la base de datos.

2. Diagrama entidad-relación

Modelo conceptual de la base de datos, respetando la nomenclatura de Peter Chen.



3. Diagrama relacional



4. Script Completo

```

CREATE DOMAIN MaximoPoblacionPlaneta AS BIGINT
CHECK (VALUE <= 99999999);
CREATE DOMAIN MaximoNombreSistema AS VARCHAR(30);
CREATE DOMAIN MaximoIdPlaneta AS NUMERIC(16,0)
CHECK (VALUE <=9999999999999999);

CREATE TABLE SistemaSolar
(
  IdSistemaSolar NUMERIC(20,0) PRIMARY KEY,
  NombreSistema MaximoNombreSistema NOT NULL,
  PlanetasNoHabitables INT NOT NULL,
  Soles INT NOT NULL CHECK (Soles >= 1 and Soles <= 3)
);
ALTER TABLE SistemaSolar
ADD CONSTRAINT CHK_PlanetasNoHabitables CHECK
(PlanetasNoHabitables >= 0 and PlanetasNoHabitables <=20);

CREATE TABLE Planeta
(
  IdPlaneta MaximoIdPlaneta PRIMARY KEY,
  NombreComun VARCHAR(30) NOT NULL,
  IdSistemaSolar NUMERIC(20,0) NOT NULL,
  FOREIGN KEY (IdSistemaSolar) REFERENCES SistemaSolar(IdSistemaSolar)
  ON DELETE NO ACTION ON UPDATE CASCADE
);

CREATE TABLE EstacionEspacial
(
  IdEstacion NUMERIC(12,0) PRIMARY KEY,
  NombreComunEstacion VARCHAR(30) NOT NULL,
  CantidadPoblacion BIGINT NOT NULL,
  IdPlaneta NUMERIC(15,0) NOT NULL,
  FOREIGN KEY (IdPlaneta) REFERENCES Planeta(IdPlaneta)
  ON DELETE CASCADE ON UPDATE CASCADE
);
ALTER TABLE EstacionEspacial
ADD CONSTRAINT CHK_CantidadPoblacion
CHECK (CantidadPoblacion >= 1 and CantidadPoblacion <=999999999);

CREATE TABLE Recurso
(
  NombreRecurso VARCHAR(30) NOT NULL,
  CantidadRecurso BIGINT NOT NULL CHECK (CantidadRecurso >= 1),
  IdPlaneta NUMERIC(15,0) NOT NULL,
  FOREIGN KEY (IdPlaneta) REFERENCES Planeta(IdPlaneta)

```

```
ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE Caracteristicas
(
Tamano BIGINT NOT NULL CHECK (Tamano>=1 and Tamano <= 9999999999),
CantidadPoblacion MaximoPoblacionPlaneta NOT NULL,
CantidadAgua BIGINT NOT NULL CHECK (CantidadAgua >= 1),
FechaLlegada DATE NOT NULL,
Idioma VARCHAR(30) NOT NULL,
IdPlaneta NUMERIC(15,0) PRIMARY KEY,
FOREIGN KEY (IdPlaneta) REFERENCES Planeta(IdPlaneta)
ON DELETE CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE Director
(
Nombre VARCHAR(30) NOT NULL,
IdDirector INT PRIMARY KEY,
Edad INT NOT NULL,
IdEstacion NUMERIC(12,0) NOT NULL,
FOREIGN KEY (IdEstacion) REFERENCES EstacionEspacial(IdEstacion)
ON DELETE NO ACTION ON UPDATE CASCADE
);
ALTER TABLE Director
ADD CONSTRAINT CHK_Edad CHECK (Edad >= 20 and Edad <=100);
```

5. Inserts

Como lo especifica el pdf, no es necesario agregarlos aqui, ya esta incluido en los archivos adjuntos. Pero te dejo una bonita recomendacion de album que encuentre mientras hacia esto.



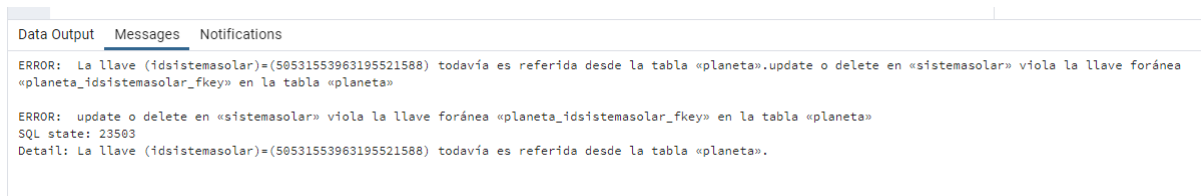
https://open.spotify.com/album/4i2103uVh5palcfFhCj1T7?si=HLkwZf7NR0qsKvr_CQYLww

6. Funcionamiento de restricciones de integridad

Evidencia del funcionamiento de al menos 4 restricciones de integridad referencial.

1. Para las tablas tenemos **SistemaSolar** y **Planeta** siendo la **FK** IdSistemaSolar y la **PK** IdPlaneta. Este trigger fue elegido, porque a mi parecer y para este ejemplo se me haría muy tonto el poder borrar un sistema solar, suponiendo que esa cosa tiene muchos mas planetas por explorar y que posiblemente no hayamos colonizado. La instrucción utilizada para esto es **ON DELETE NO ACTION**

```
DELETE FROM SistemaSolar WHERE IdSistemaSolar=50531553963195521588;
```



Como vemos, no nos dejan borrar un sistema solar. Y podemos tener seguridad que ese sistema se queda ahí.

2. Para la segunda tenemos **Planeta** y **EstacionEspacial** siendo **FK** IdPlaneta y **PK** IdEstacion. Este trigger fue elegido para actualizar los datos de un UPDATE de planeta, en específico si cambia su ID, ya que es donde perdería conexión con esta tabla y con otras. Utilizamos **ON UPDATE CASCADE**

```
UPDATE Planeta SET IdPlaneta= 13147277772978  
WHERE IdPlaneta=131472088872978;
```

Data Output Messages Notifications				
	idestacion [PK] numeric (12)	nombrecomunestacion character varying (30)	cantidadpoblacion bigint	idplaneta numeric (15)
1	335523893038	Matsoft	11814239	13147277772978

(a)

Data Output Messages Notifications			
	idplaneta [PK] numeric (15)	nombrecomun character varying (30)	idsistemasolar numeric (20)
1	13147277772978	Zamit	50531553963195521588

(b)

Podemos ver como se actualiza.

3. Ahora haremos el otro caso, que pasa, podemos llegar al punto de un planeta en que nos acabamos todo, y no queda mas que irnos de ahí y en nuestra base de datos no nos sirve un planeta que no tenga nada, entonces toca borrarlo y necesitamos que todas las referencias a este se borren. Tenemos como **FK** IdPlaneta y como **PK** IdEstacion. Pero no te preocupes esto funciona para todas las tablas en donde hacen referencia a un IdPlaneta. **ON DELETE CASCADE**.

```
DELETE FROM Planeta WHERE IdPlaneta= 13147277772978;
```

Data Output Messages Notifications		
	idplaneta [PK] numeric (15)	idsistemasolar numeric (20)

(a)

Data Output Messages Notifications			
	idestacion [PK] numeric (12)	nombrecomunestacion character varying (30)	idplaneta numeric (15)

(b)

4. Por ultimo una restriccion de **ON DELETE NO ACTION** pero para la estacion espacial, ya que no tiene mucho sentido borrar una estacion espacial, si cada estacion es construida a partir de un planeta, entonces si se borra una estacion no tendríamos ”forma de comunicarnos con el planeta”, entonces como **Fk** tenemos a IdEstacion y como **PK** tenemos IdDirector.

```
DELETE FROM EstacionEspacial WHERE IdEstacion= 335523893038;
```

Data Output	Messages	Notifications
<p>ERROR: La llave (idestacion)=(335523893038) todavía es referida desde la tabla «director».update o delete en «estacionespacial» viola la llave foránea «director_idestacion_fkey» en la tabla «director»</p> <p>ERROR: update o delete en «estacionespacial» viola la llave foránea «director_idestacion_fkey» en la tabla «director»</p> <p>SQL state: 23503</p> <p>Detail: La llave (idestacion)=(335523893038) todavía es referida desde la tabla «director».</p>		

Con esto tenemos la seguridad de que no borraremos una estación asociada a un planeta ya que es uno a uno la relación.

7. Funcionamiento de restricciones check

Evidencia del funcionamiento de al menos 3 restricciones check para “atributos” de varias tablas.

1. Para la primera elegimos la tabla SistemaSolar, esto porque pues es la primera y por poner un check, elegimos el atributo de planetasNoHabitables, esto porque pues se me ocurre que a nosotros del futuro se nos complicaría ir a una sistema solar donde haya en su mayoría planetas no habitables y que hagan que las colonizaciones sean mucho mas complicadas entonces por ello es que ponemos un máximo de 20 planetas y que empieza a contar de 0.

Instruccion

```
ALTER TABLE SistemaSolar
ADD CONSTRAINT CHK_PlanetasNoHabitables
CHECK (PlanetasNoHabitables >= 0 and PlanetasNoHabitables <=20);
```

Evidencia del funcionamiento

```
UPDATE SistemaSolar SET PlanetasNoHabitables=30
WHERE IdSistemaSolar=50531553963195521588;
```

Data Output	Messages	Notifications
ERROR: La fila que falla contiene (50531553963195521588, 91L33462oqV3g, 30, 1).el nuevo registro para la relación «sistemasolar» viola la restricción «check» «chk_planetasnohabitables»		
ERROR: el nuevo registro para la relación «sistemasolar» viola la restricción «check» «chk_planetasnohabitables»		
SQL state: 23514		
Detail: La fila que falla contiene (50531553963195521588, 91L33462oqV3g, 30, 1).		

2. Para la segunda elegimos a la tabla EstacionEspacial y es para controlar la poblacion que hay, ya que en la estacion hay un maximo de gente que puede estar.

Instruccion

```
ALTER TABLE EstacionEspacial
ADD CONSTRAINT CHK_CantidadPoblacion
CHECK (CantidadPoblacion >= 1 and CantidadPoblacion <=99999999);
```

Evidencia del funcionamiento

```
UPDATE EstacionEspacial SET CantidadPoblacion=199999999
WHERE IdEstacion=335523893038;
```

Data Output	Messages	Notifications
ERROR: La fila que falla contiene (335523893038, Matsoft, 199999999, 131472088872978).el nuevo registro para la relación «estacionespacial» viola la restricción «check» «chk_cantidadpoblacion»		
ERROR: el nuevo registro para la relación «estacionespacial» viola la restricción «check» «chk_cantidadpoblacion»		
SQL state: 23514		
Detail: La fila que falla contiene (335523893038, Matsoft, 199999999, 131472088872978).		

3. Para el ultimo elegimos la tabla Director y esto para ver que nuestro director cumpla como requisito tener por lo menos 20 años y como máximo 100.

Instruccion

```
ALTER TABLE Director
ADD CONSTRAINT CHK_Edad
CHECK (Edad >= 20 and Edad <=100);
```

Evidencia del funcionamiento

```
UPDATE Director SET Edad=19
WHERE IdDirector=967731965;
```

Data Output	Messages	Notifications
ERROR: La fila que falla contiene (Zarla Kingaby, 967731965, 19, 405764651419).el nuevo registro para la relación «director» viola la restricción «check» «chk_edad»		
ERROR: el nuevo registro para la relación «director» viola la restricción «check» «chk_edad»		
SQL state: 23514		
Detail: La fila que falla contiene (Zarla Kingaby, 967731965, 19, 405764651419).		

8. Dominios personalizados

Evidencia de la creación de al menos tres dominios personalizados. Se deben utilizar restricciones check en la creación de los tres dominios.

1. Para el primer dominio elegimos a la tabla SistemaSolar, en este caso al atributo de nombre, haciendo que su restricción sea en cuanto a máximo 30 caracteres.

```
CREATE DOMAIN MaximoNombreSistema AS VARCHAR(30);
```

La modificacion de la tabla es la siguiente.

```
CREATE TABLE SistemaSolar
(
  IdSistemaSolar NUMERIC(20,0) PRIMARY KEY,
  NombreSistema MaximoNombreSistema NOT NULL,
  PlanetasNoHabitables INT NOT NULL,
  Soles INT NOT NULL
);
```

Lo probamos con la siguiente instrucción.

```
UPDATE SistemaSolar SET NombreSistema=
ElSistemaMasChidoYBonitoDeTodasLasJodidasGalaxiasYaceAqui
WHERE IdSistemaSolar = 50531553963195521588;
```

Data Output	Messages	Notifications
ERROR: el valor es demasiado largo para el tipo character varying(30)		
SQL state: 22001		

2. Para el segundo dominio elegimos la tabla Caracteristicas, en este caso elegimos el atributo CantidadPoblacion, Dando un máximo de población para cada planeta.

```
CREATE DOMAIN MaximoPoblacionPlaneta
AS BIGINT CHECK (VALUE <= 99999999);
```

La modificación en la tabla es la siguiente.

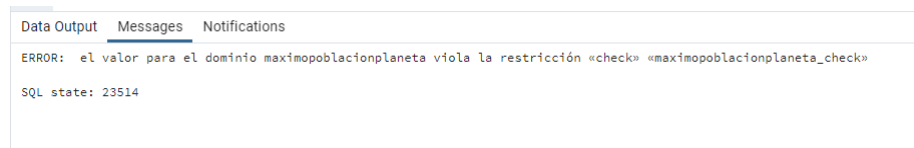
```
CREATE TABLE Caracteristicas
(
```



```
Tamano BIGINT NOT NULL,
CantidadPoblacion MaximoPoblacionPlaneta NOT NULL,
CantidadAgua BIGINT NOT NULL,
FechaLlegada DATE NOT NULL,
Idioma VARCHAR(30) NOT NULL,
IdPlaneta NUMERIC(15,0) PRIMARY KEY,
FOREIGN KEY (IdPlaneta) REFERENCES Planeta(IdPlaneta)
ON DELETE CASCADE ON UPDATE CASCADE
);
```

Lo probamos con la siguiente instrucción.

```
UPDATE Caracteristicas SET CantidadPoblacion = 199999999
WHERE IdPlaneta= 131472088872978;
```



3. Para el ultimo dominio Usaremos la tabla Planeta y elegimos el id, para darle un máximo de dígitos que puede tener.

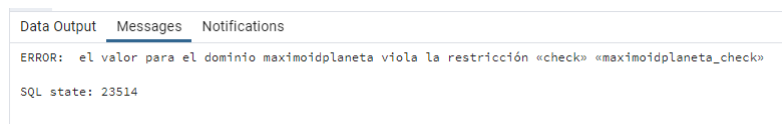
```
CREATE DOMAIN MaximoIdPlaneta AS NUMERIC(16,0)
CHECK (VALUE <=9999999999999999);
```

La modificación en la tabla es la siguiente.

```
CREATE TABLE Planeta
(
IdPlaneta MaximoIdPlaneta PRIMARY KEY,
NombreComun VARCHAR(30) NOT NULL,
IdSistemaSolar NUMERIC(20,0) NOT NULL,
FOREIGN KEY (IdSistemaSolar) REFERENCES SistemaSolar(IdSistemaSolar)
ON DELETE NO ACTION ON UPDATE CASCADE
);
```

Lo probamos con la siguiente instrucción.

```
UPDATE Planeta SET IdPlaneta = 1999999999999999
WHERE IdPlaneta = 699376528536750;
```



9. Restricciones para tuplas

Evidencia del funcionamiento de al menos 2 restricciones para “tuplas” en diferentes tablas (Unidad 8 Integridad, tema “Specifying Constraints on Tuples Using CHECK”)

1. Para la primera Elegimos la tabla sistema solar y vamos a Soles, en específico creo que es recomendable que no haya tantos soles en el sistema, y lo ponemos como menor igual a 3 soles y mayor igual a 1 sol porque necesitamos de un solesito para la vida.

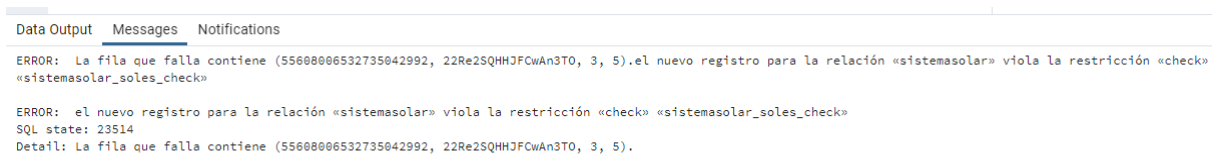
```
CHECK (Soles >= 1 and Soles <= 3)
```

Así quedaría nuestro CHECK en la tabla

```
CREATE TABLE SistemaSolar
(
  IdSistemaSolar NUMERIC(20,0) PRIMARY KEY,
  NombreSistema MaximoNombreSistema NOT NULL,
  PlanetasNoHabitables INT NOT NULL,
  Soles INT NOT NULL CHECK (Soles >= 1 and Soles <= 3)
);
```

Probamos con:

```
UPDATE SistemaSolar SET Soles=5
WHERE IdSistemaSolar =55608006532735042992;
```



The screenshot shows the 'Messages' tab in SQL Server Enterprise Manager. It displays an error message: 'ERROR: La fila que falla contiene (55608006532735042992, 22Re25QHHJFCwAn3T0, 3, 5).el nuevo registro para la relación «sistemasolar» viola la restricción «check» «sistemasolar_soles_check»'. Below this, it says 'ERROR: el nuevo registro para la relación «sistemasolar» viola la restricción «check» «sistemasolar_soles_check»', 'SQL state: 23514', and 'Detail: La fila que falla contiene (55608006532735042992, 22Re25QHHJFCwAn3T0, 3, 5)'.

2. Para igual evitar incluir planetas donde no haya agua o simplemente se fue un menos, declaramos que por lo menos haya 1 litro de agua. Ya con esto podemos estar seguros que en cada planeta hay agua.

```
CHECK (CantidadAgua >=1);
```

Así quedaría en nuestra tabla

```
CREATE TABLE Caracteristicas
(
  Tamano BIGINT NOT NULL
  CHECK (Tamano>=1 and Tamano <= 9999999999),
  CantidadPoblacion MaximoPoblacionPlaneta NOT NULL,
  CantidadAgua BIGINT NOT NULL CHECK (CantidadAgua >= 1),
  FechaLlegada DATE NOT NULL,
  Idioma VARCHAR(30) NOT NULL,
  IdPlaneta NUMERIC(15,0) PRIMARY KEY,
  FOREIGN KEY (IdPlaneta) REFERENCES Planeta(IdPlaneta)
  ON DELETE CASCADE ON UPDATE CASCADE
);
```

Probamos con:

```
UPDATE Caracteristicas SET CantidadAgua =-2
WHERE IdPlaneta = 489691046467286;
```

Data Output	Messages	Notifications
ERROR: La fila que falla contiene (2126603872, 29790685, -2, 2088-07-30, Catalan, 489691046467286).el nuevo registro para la relación «características» viola la restricción «check» «características_cantidadagua_check»		
ERROR: el nuevo registro para la relación «características» viola la restricción «check» «características_cantidadagua_check» SQL state: 23514 Detail: La fila que falla contiene (2126603872, 29790685, -2, 2088-07-30, Catalan, 489691046467286).		

10. Consultas

Plantea 3 consultas que consideres relevantes para la base de datos propuesta.

1. Como primera consulta obtendremos el nombre del sistema solar, la cantidad de planetas no habitables en este sistema, al igual que la información de los planetas habitables de los sistemas. Ojo, Esta consulta sirve para un unico sistema, es decir que nosotros digitamos el id y nos muestra toda esa informacion.

```
SELECT
    ss.NombreSistema AS SistemaSolar ,
    ss.PlanetasNoHabitables ,
    p.IdPlaneta ,
    p.NombreComun ,
    c.Tamano ,
    c.CantidadPoblacion ,
    c.CantidadAgua
FROM SistemaSolar ss
JOIN Planeta p ON ss.IdSistemaSolar = p.IdSistemaSolar
JOIN Caracteristicas c ON p.IdPlaneta = c.IdPlaneta
WHERE ss.IdSistemaSolar = IdSistemaSolarDigitado;
```

	sistemasolar character varying (30)	planetasnohabitables integer	idplaneta numeric (16)	nombrecomun character varying (30)	tamaño bigint	cantidadpoblacion bigint	cantidadagua bigint
1	9iL33462oqV3g	11	131472088872978	Zamit	8591903186	51064284	34329083347918
2	9iL33462oqV3g	11	417975357685977	Sonsing	4195118070	80863302	20537999977678

2. La segunda consulta es para obtener la lista de estaciones espaciales y sus directores ordenados por la cantidad de población de manera ascendente.

```
SELECT
    e.NombreComunEstacion AS EstacionEspacial ,
    e.CantidadPoblacion ,
    d.Nombre AS NombreDirector ,
    d.Edad
FROM EstacionEspacial e
JOIN Director d ON e.IdEstacion = d.IdEstacion
ORDER BY e.CantidadPoblacion DESC;
```

3. Para la ultima consulta queremos saber que planetas, y recursos que tienen, tanto el nombre como la cantidad. Todo esto a traves del Id del sistema solar.

```
SELECT
```

Data Output Messages Notifications				
	estacionespacial character varying (30)	cantidadpoblacion bigint	nombredirector character varying (30)	edad integer
1	Zamit	99524056	Maude Shearme	84
2	Job	99520803	Sybilla Lisamore	20
3	Regrant	96735610	Klarrisa Dankersley	67
4	Cookley	96658848	Arielle Jessup	40
5	Daltfresh	96452739	Zaneta Cristofano	46
6	Zamit	96043985	Delaney Stainburn	92
7	Tempsoft	95693232	Carline Hazeldean	93
8	Keylex	95658496	Olivette Wendover	88
9	Cardify	95448142	Tamra Fruish	23
10	Greenlam	94524842	Rollins Ruller	98
11	Sonair	93580128	Caty Cumine	70
12	Y-find	93380240	Celka Ferretti	91
13	Solarbreeze	92692466	Berthe Thickers	78
Total rows: 199 of 199		Query complete 00:00:00.102		

```

r.NombreRecurso ,
r.CantidadRecurso ,
p.NombreComun AS NombrePlaneta
FROM Recurso r
JOIN Planeta p ON r.IdPlaneta = p.IdPlaneta
JOIN SistemaSolar ss ON p.IdSistemaSolar = ss.IdSistemaSolar
WHERE ss.IdSistemaSolar = idquetuquieraspapitochulo;

```

Data Output Messages Notifications			
	nombrecurso character varying (30)	cantidadrecurso bigint	nombrepianeta character varying (30)
1	Diamante	78989615	Trippledex
2	Diamante	47928653	Tresom

11. Vistas

Plantea 3 vistas que consideres relevantes para la base de datos propuesta.

1. Para la primera vista tenemos que queremos ver a los planetas y principalmente sus recursos, además de agregar las características de cada planeta.

```

CREATE VIEW VistaPlanetasDetallada AS
SELECT
p.IdPlaneta ,
p.NombreComun ,
c.Tamanio ,
c.CantidadPoblacion ,
c.CantidadAgua ,
r.NombreRecurso ,
r.CantidadRecurso
FROM Planeta p
JOIN Caracteristicas c ON p.IdPlaneta = c.IdPlaneta
LEFT JOIN Recurso r ON p.IdPlaneta = r.IdPlaneta
ORDER BY c.CantidadPoblacion DESC;

```

Data Output

Messages

Notifications

	idplaneta numeric (16)	nombrecomun character varying (30)	tamaño bigint	cantidadpoblacion bigint	cantidadagua bigint	nombrerecurso character varying (30)	cantidadrecurso bigint
1	157333652457522	Daltfresh	90201518	99643892	3400865990755	Carbon	28167293
2	705784772727564	Duobam	4907244785	99476210	43367297138422	Titanio	24346356
3	678196356265232	Overhold	1130921315	99038618	74849541044481	Carne	94792868
4	365461385950645	Stim	9945993532	98483582	39759938501255	Madera	32688836
5	501719133857549	Viva	3197050293	98321948	62496646374600	Carne	85833003
6	427761541531180	Cookieley	9151060995	97811638	8938505412006	Madera	54014133
7	297111325809910	Asoka	7666679708	97477814	37848461621978	Hierro	74929287
8	861572655909092	Tempsoft	5317118235	97393388	74825016612834	Titanio	38662815
9	116338685731271	Fintone	3239680674	97333662	48126448486401	Diamante	55499706
10	361850101713385	Asoka	6502376157	96540297	71072886350703	Carbon	86369903
11	211079125599641	Job	6534798896	95574629	80929235683834	Azufre	17693906
12	461761046527655	Fintone	4077855946	95247141	88933322989710	Titanio	99840970
13	996057079728469	Zathin	6394032851	95143899	50685272895938	Azufre	25208801
Total rows: 198 of 198		Query complete 00:00:00.175					

Ejemplo de uso.

```
SELECT * FROM VistaPlanetasDetallada ;
```

Ejemplo de uso. Este busca en especifico un Sistema Solar y te muestra información detallada los planetas a través de su nombre.

```
SELECT *
FROM VistaPlanetasDetallada
WHERE IdPlaneta IN (SELECT IdPlaneta FROM Planeta
WHERE IdSistemaSolar = (SELECT IdSistemaSolar FROM SistemaSolar WHERE NombreSistemaSolar = 'Sistema Solar'))
```

- Para la segunda vista daremos las listas de de los sistemas solares y Los recursos que este dispone, Mostrando únicamente el Sistema, el id, nombre y los recursos que tiene.

```
CREATE VIEW VistaSistemasSolaresConRecursos AS
SELECT
ss.IdSistemaSolar ,
ss.NombreSistema ,
SUM(CASE WHEN r.NombreRecurso = 'Diamante' THEN r.CantidadRecurso END)
AS Diamante ,
SUM(CASE WHEN r.NombreRecurso = 'Azufre' THEN r.CantidadRecurso END)
AS Azufre ,
SUM(CASE WHEN r.NombreRecurso = 'Carne' THEN r.CantidadRecurso END)
AS Carne ,
SUM(CASE WHEN r.NombreRecurso = 'Hierro' THEN r.CantidadRecurso END)
AS Hierro ,
SUM(CASE WHEN r.NombreRecurso = 'Madera' THEN r.CantidadRecurso END)
AS Madera ,
SUM(CASE WHEN r.NombreRecurso = 'Platino' THEN r.CantidadRecurso END)
AS Platino ,
SUM(CASE WHEN r.NombreRecurso = 'Titania' THEN r.CantidadRecurso END)
AS Titania ,
SUM(CASE WHEN r.NombreRecurso = 'Arena' THEN r.CantidadRecurso END)
AS Arena ,
SUM(CASE WHEN r.NombreRecurso = 'Carbon' THEN r.CantidadRecurso END)
```

```

AS Carbon
FROM SistemaSolar ss
LEFT JOIN Planeta p ON ss.IdSistemaSolar = p.IdSistemaSolar
LEFT JOIN Recurso r ON p.IdPlaneta = r.IdPlaneta
GROUP BY ss.IdSistemaSolar , ss.NombreSistema;

```

Data Output Messages Graph Visualiser x Notifications													
	idsistemasolar numeric (20)	nombresistema character varying (30)	diamante numeric	azufre numeric	carne numeric	hierro numeric	madera numeric	platino numeric	titanio numeric	arena numeric	carbon numeric		
1	520817003907501527	wCclnpS	[null]	[null]	[null]	[null]	[null]	76129432	[null]	[null]	60622656		
2	60986538460297555234	be9euMD	[null]	99452979	[null]	[null]	[null]	56056235	[null]	[null]	[null]		
3	40248563556756667321	1464	13873570	[null]	[null]	[null]	[null]	[null]	[null]	[null]	[null]		
4	99442069778410448582	Xqq366EsqcyIWtVUnMpl6F9vrWv3TK	[null]	[null]	[null]	[null]	[null]	72317481	[null]	[null]	1608021		
5	5535725369514945655	S	[null]	[null]	[null]	[null]	[null]	[null]	108861483	[null]	[null]		
6	35713118431267961797	4fyhqQB1f05FQp6BQvOud	[null]	[null]	[null]	[null]	61254278	39392136	[null]	[null]	[null]		
7	27469541653369489540	4766r	[null]	[null]	[null]	[null]	[null]	[null]	99840970	47167415	[null]		
8	36009495037493587469	lb	[null]	[null]	32953585	[null]	[null]	[null]	[null]	[null]	16176513		
9	13106833943549074361	O37yV9rORo2QvLJpu	[null]	[null]	98709527	[null]	[null]	[null]	[null]	[null]	26501551		
10	38436211997335356968	SATckIUjMSIqu9j6AqiE414S0ZME19	[null]	[null]	138108655	[null]	[null]	[null]	[null]	[null]	[null]		
11	90156519704538682211	38975mLc	6654476	[null]	51068791	[null]	[null]	[null]	[null]	[null]	[null]		
12	37480428774418160150	5	[null]	[null]	16482902	[null]	[null]	[null]	4300974	[null]	[null]		
13	30711025693944845936	M88z6057mxEHLB5jsVU8G7ME3Z	32876487	[null]	[null]	73841536	[null]	[null]	[null]	[null]	[null]		
Total rows: 100 of 100 Query complete 00:00:00.077													

Ejemplo de uso

```
SELECT * FROM VistaSistemasSolaresConRecursos;
```

Otro ejemplo de uso. Este es un poco trampa, porque usa la vista de abajo para dar información mas detallada.

```

SELECT vcp.IdPlaneta , vcp.NombreComun, vcp.ClasificacionPoblacion , vcsr.*
FROM VistaClasificacionPoblacion vcp
JOIN VistaSistemasSolaresConRecursos vcsr ON vcp.IdPlaneta = vcsr.IdPlaneta;

```

- Para nuestra tercera vista algo que podria ayudarnos mucho al ver los planetas es verlo por una cantidad de poblacion, pero que sea una columna aparte, que apartir de cierto numero nos indique que esta muy poblado o moderadamente poblado o poco poblado.

```

CREATE VIEW VistaClasificacionPoblacion AS
SELECT
p.IdPlaneta ,
p.NombreComun,
SUM(c.CantidadPoblacion) AS TotalPoblacion ,
CASE
WHEN SUM(c.CantidadPoblacion) < 1000000 THEN 'Poco Poblado'
WHEN SUM(c.CantidadPoblacion) >= 1000000 AND SUM(c.CantidadPoblacion)
< 10000000 THEN 'Moderadamente Poblado'
WHEN SUM(c.CantidadPoblacion) >= 10000000 THEN 'Altamente Poblado'
END AS ClasificacionPoblacion
FROM Planeta p
JOIN Caracteristicas c ON p.IdPlaneta = c.IdPlaneta

```

GROUP BY p.IdPlaneta , p.NombreComun;

Data Output Messages Graph Visualiser × Notifications				
<div> <div> <div>≡</div> <div>+</div> </div> <div> <div>📄</div> <div>▼</div> </div> <div> <div>📋</div> <div>▼</div> </div> <div> <div>🗑️</div> </div> <div> <div>📦</div> <div>⬇️</div> </div> <div> <div>📈</div> </div> </div>				
	idplaneta numeric (16)	nombrecomun character varying (30)	totalpoblacion numeric	clasificacionpoblacion text
1	673605923362750	Ventosanzap	94938219	Altamente Poblado
2	584076474789282	Tin	75184416	Altamente Poblado
3	211079125599641	Job	95574629	Altamente Poblado
4	785223110675391	Voyatouch	8900246	Moderadamente Poblado
5	524825975696811	Sonair	19526055	Altamente Poblado
6	729029643561456	Bitchip	24456275	Altamente Poblado
7	575292073692182	Zoolab	48880890	Altamente Poblado
8	466242825526681	Bitwolf	63443237	Altamente Poblado
9	163887016455738	Alpha	26529123	Altamente Poblado
10	938534836218619	Holdlamis	35169609	Altamente Poblado
11	807820001762800	Namfix	87189006	Altamente Poblado
12	352180317938868	Bytecard	13581395	Altamente Poblado
13	334954890671719	Fintone	73771391	Altamente Poblado
Total rows: 198 of 198		Query complete 00:00:00.207		

Ejemplo de uso

```
SELECT * FROM VistaClasificacionPoblacion;
```

Ejemplo de uso. Este sirve para filtrar con poblaciones

```
SELECT *
FROM VistaClasificacionPoblacion
WHERE ClasificacionPoblacion = 'Altamente Poblado';
```

Aclaraciones

Solo una de las tablas tiene 100 registros, las demás tienen 200, esto porque se me hacia medio aburrido hacer 1 sistema solar y que aquí solo hubiera 1 planeta, entonces distribuí de a 2 planetas por sistema solar en el insert, que claro ya agregando mas obvio es mas divertido ver los resultados, pero ya es mucho, dure mucho time en el insert :c, esto mientras que se cumplan las restricciones. pero por ello es que solo en la tabla de SistemaSolar hay 100 registros y en las demás 200. Pero funciona de igual forma con 100 o menos, solo fue para hacerlo un poco mas visible el como funcionaba el sistema solar.