CREATE DATABASE universidad;  
  
USE universidad;  
  
CREATE TABLE estudiantes  
(  
 id\_est INTEGER AUTO\_INCREMENT PRIMARY KEY NOT NULL,  
 nombres VARCHAR(50),  
 apellidos VARCHAR(50),  
 edad INTEGER,  
 gestion INTEGER,  
 fono INTEGER,  
 email VARCHAR(100),  
 direccion VARCHAR(100),  
 sexo VARCHAR(10)  
);  
  
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)  
 VALUES ('Miguel' ,'Gonzales Veliz', 20, 2832115, 'miguel@gmail.com', 'Av. 6 de Agosto', 'masculino');  
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)  
 VALUES ('Sandra' ,'Mavir Uria', 25, 2832116, 'sandra@gmail.com', 'Av. 6 de Agosto', 'femenino');  
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)  
 VALUES ('Joel' ,'Adubiri Mondar', 30, 2832117, 'joel@gmail.com', 'Av. 6 de Agosto', 'masculino');  
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)  
 VALUES ('Andrea' ,'Arias Ballesteros', 21, 2832118, 'andrea@gmail.com', 'Av. 6 de Agosto', 'femenino');  
INSERT INTO estudiantes (nombres, apellidos, edad, fono, email, direccion, sexo)  
 VALUES ('Santos' ,'Montes Valenzuela', 24, 2832119, 'santos@gmail.com', 'Av. 6 de Agosto', 'masculino');  
  
SELECT est.\*  
FROM estudiantes AS est;  
  
CREATE TABLE materias  
(  
 id\_mat INTEGER AUTO\_INCREMENT PRIMARY KEY NOT NULL,  
 nombre\_mat VARCHAR(100),  
 cod\_mat VARCHAR(100)  
);  
  
CREATE TABLE inscripcion  
(  
 id\_ins INTEGER AUTO\_INCREMENT PRIMARY KEY NOT NULL,  
 id\_est INT NOT NULL,  
 id\_mat INT NOT NULL,  
 semestre VARCHAR(20),  
 gestion INTEGER,  
 FOREIGN KEY (id\_est) REFERENCES estudiantes (id\_est),  
 FOREIGN KEY (id\_mat) REFERENCES materias (id\_mat)  
);  
  
INSERT INTO materias (nombre\_mat, cod\_mat) VALUES ('Introduccion a la Arquitectura','ARQ-101');  
INSERT INTO materias (nombre\_mat, cod\_mat) VALUES ('Urbanismo y Diseno','ARQ-102');  
INSERT INTO materias (nombre\_mat, cod\_mat) VALUES ('Dibujo y Pintura Arquitectonico','ARQ-103');  
INSERT INTO materias (nombre\_mat, cod\_mat) VALUES ('Matematica discreta','ARQ-104');  
INSERT INTO materias (nombre\_mat, cod\_mat) VALUES ('Ingenieria Sistemas','SIS-121');  
  
INSERT INTO inscripcion (id\_est, id\_mat, semestre, gestion) VALUES (1, 1, '1er Semestre', 2015);  
INSERT INTO inscripcion (id\_est, id\_mat, semestre, gestion) VALUES (1, 2, '2do Semestre', 2015);  
INSERT INTO inscripcion (id\_est, id\_mat, semestre, gestion) VALUES (2, 4, '1er Semestre', 2016);  
INSERT INTO inscripcion (id\_est, id\_mat, semestre, gestion) VALUES (2, 3, '2do Semestre', 2016);  
INSERT INTO inscripcion (id\_est, id\_mat, semestre, gestion) VALUES (3, 3, '2do Semestre', 2017);  
INSERT INTO inscripcion (id\_est, id\_mat, semestre, gestion) VALUES (3, 1, '3er Semestre', 2017);  
INSERT INTO inscripcion (id\_est, id\_mat, semestre, gestion) VALUES (4, 4, '4to Semestre', 2017);  
INSERT INTO inscripcion (id\_est, id\_mat, semestre, gestion) VALUES (5, 5, '5to Semestre', 2017);  
  
SELECT est.\*  
FROM estudiantes AS est;  
  
SELECT mat.\*  
FROM materias AS mat;  
  
SELECT ins.\*  
FROM inscripcion AS ins;  
  
# Mostrar nombres y apellidos de los estudiantes inscritos de la materia ARQ-104  
  
SELECT es.id\_est, es.nombres, es.apellidos, mat.nombre\_mat, mat.cod\_mat  
FROM estudiantes AS es  
INNER JOIN inscripcion AS ins ON es.id\_est = ins.id\_est  
INNER JOIN materias AS mat ON ins.id\_mat = mat.id\_mat  
WHERE mat.cod\_mat = 'ARQ-104';  
  
SELECT count(est.id\_est)  
FROM estudiantes AS est;  
  
SELECT AVG(est.edad)  
FROM estudiantes AS est;  
  
SELECT MAX(est.edad)  
FROM estudiantes AS est;  
  
SELECT MIN(est.edad)  
FROM estudiantes AS est;  
  
SELECT ins.id\_est, MAX(ins.id\_mat)  
FROM inscripcion AS ins  
GROUP BY ins.id\_est;  
  
SELECT ins.id\_est, AVG(ins.id\_mat)  
FROM inscripcion AS ins  
GROUP BY ins.id\_est;  
  
SELECT ins.id\_est, MIN(ins.id\_mat)  
FROM inscripcion AS ins  
GROUP BY ins.id\_est;  
  
SELECT ins.id\_est, SUM(ins.id\_mat)  
FROM inscripcion AS ins  
GROUP BY ins.id\_est;  
  
SELECT ins.id\_est, avg(ins.id\_mat)  
FROM inscripcion AS ins  
GROUP BY ins.id\_est;  
  
SELECT ins.id\_est, MAX(ins.id\_mat)  
FROM inscripcion AS ins  
GROUP BY ins.id\_est;  
  
DROP FUNCTION IF EXISTS max\_edad\_estudiantes;  
  
CREATE FUNCTION *max\_edad\_estudianes*() RETURNS int  
BEGIN  
 return  
 (  
 SELECT min(est.edad)  
 FROM estudiantes AS est  
 );  
END;  
  
SELECT @max\_edad\_estudiantes;  
  
# Creando la funcion en MySQL  
# Minima edad de estudiantes  
  
CREATE FUNCTION *min\_edad\_estudianes*() RETURNS int  
BEGIN  
 return  
 (  
 SELECT min(est.edad)  
 FROM estudiantes AS est  
 );  
END;  
  
SELECT *min\_edad\_estudianes*() as minEdad;  
  
#Edadd promedio de estudiantes  
CREATE FUNCTION *promedio\_edad\_estudianes*() RETURNS int  
BEGIN  
 return  
 (  
 SELECT avg(est.edad)  
 FROM estudiantes AS est  
 );  
END;  
  
SELECT *promedio\_edad\_estudianes*() as promedioEdad;  
  
  
  
#maxima edad de sexo femenino  
  
  
#maxima edad de sexo femenino o masculino  
CREATE FUNCTION *max\_edad\_est\_2maculino*(parSexo VARCHAR(20))  
RETURNS INTEGER  
BEGIN  
 return  
 (  
 SELECT max(est.edad)  
 FROM estudiantes AS est  
 where est.sexo = parSexo  
 );  
END;  
  
select *max\_edad\_est\_2maculino*('masculino');  
  
CREATE FUNCTION *max\_edad\_est\_4maculino*(parSexo VARCHAR(20),eda integer)  
RETURNS INTEGER  
BEGIN  
 return  
 (  
 SELECT max(est.edad)  
 FROM estudiantes AS est  
 where est.sexo = parSexo AND est.edad > eda  
 );  
END;  
select *max\_edad\_est\_4maculino*('masculino',23);  
  
  
 #mostrar todos los registros de la tabla estudiantes (nombre y apellidos)  
 # si la suma de las edades de los estudiantes masculino sea par.  
CREATE FUNCTION *sum\_edad*()  
BEGIN  
 return  
 (  
 SELECT SUM(est.id\_est)  
 FROM estudiantes AS est  
 );  
END;  
CREATE or replace FUNCTION *SUM\_edad* (genero varchar(20))  
RETURNS INTEGER  
BEGIN  
 return  
 (  
 SELECT SUM(est.id\_est)  
 FROM estudiantes AS est  
 WHERE est.sexo = genero  
 );  
END;  
  
SELECT est.nombres,est.apellidos  
 FROM estudiantes AS est  
 where *SUM\_edad*('masculino') % 2 = 0;  
  
#crear una funcion que retorne una cadena  
#El mensaje podria decir "Hola mundo desde DBA"  
create function *retorna\_cadena*()  
returns varchar(100)  
begin  
 return 'Hola mundo desde DBA II';  
end;  
select *retorna\_cadena*();  
  
# Determinar la edad mayor del genero (femenino o masculino),  
# de aquellos estudiantes que sean mayores a cierta edad.  
  
CREATE or replace FUNCTION *max\_edad\_est*(edad integer, genero varchar(20))  
RETURNS INTEGER  
BEGIN  
 return  
 (  
 SELECT max(est.edad)  
 FROM estudiantes AS est  
 where est.sexo = genero and est.edad >edad  
 );  
END;  
SELECT *max\_edad\_est*(18, 'masculino');  
  
CREATE FUNCTION *max\_edad\_est\_femenino*()  
RETURNS int  
BEGIN  
 return  
 (  
 SELECT max(est.edad)  
 FROM estudiantes AS est  
 where est.sexo ='femenino'  
 );  
END;  
  
SELECT *max\_edad\_est\_femenino*() as maxfemeninoEdad;  
  
SELECT est.nombres, est.apellidos  
FROM estudiantes AS est  
where est.id\_est=1;  
  
CREATE or replace FUNCTION *max\_id*()  
RETURNS INTEGER  
BEGIN  
 return  
 (  
 SELECT MAX(est.id\_est)  
 FROM estudiantes AS est  
 );  
END;  
SELECT est.id\_est,est.nombres,est.apellidos  
 FROM estudiantes AS est  
 where est.id\_est = *max\_id*();  
  
# Determinar la edad mayor del genero (femenino o masculino),  
# de aquellos estudiantes que sean mayores a cierta edad.  
  
CREATE or replace FUNCTION *max\_edad\_est*(edad integer, genero varchar(20))  
RETURNS INTEGER  
BEGIN  
 return  
 (  
 SELECT max(est.edad)  
 FROM estudiantes AS est  
 where est.sexo = genero and est.edad >edad  
 );  
END;  
SELECT *max\_edad\_est*(18, 'masculino');  
  
CREATE FUNCTION *max\_edad\_est\_femenino*()  
RETURNS int  
BEGIN  
 return  
 (  
 SELECT max(est.edad)  
 FROM estudiantes AS est  
 where est.sexo ='femenino'  
 );  
END;  
  
SELECT *max\_edad\_est\_femenino*() as maxfemeninoEdad;  
  
SELECT est.nombres, est.apellidos  
FROM estudiantes AS est  
where est.id\_est=1;  
  
CREATE or replace FUNCTION *max\_id*()  
RETURNS INTEGER  
BEGIN  
 return  
 (  
 SELECT MAX(est.id\_est)  
 FROM estudiantes AS est  
 );  
END;  
SELECT est.nombres,est.apellidos  
 FROM estudiantes AS est  
 where est.id\_est = *max\_id*();  
  
create or replace function *sum\_edad*(genero varchar(20))  
returns integer  
begin  
 return (  
 select sum(est.edad)  
 from estudiantes as est  
 where est.sexo=genero  
 );  
end;  
  
SELECT est.nombres,est.apellidos  
 FROM estudiantes AS est  
 where *sum\_edad*('masculino') % 2 = 0;  
  
CREATE or replace FUNCTION *min\_edad\_est*(edad integer, genero varchar(20))  
RETURNS INTEGER  
BEGIN  
 return  
 (  
 SELECT min(est.edad)  
 FROM estudiantes AS est  
 where est.sexo = genero and est.edad >edad  
 );  
END;  
SELECT *min\_edad\_est*(18, 'femenino');  
  
CREATE or replace FUNCTION *inscritos*()  
RETURNS int  
BEGIN  
 return  
 (  
 SELECT (ins.id\_mat)  
 FROM inscripcion AS ins  
 where ins.id\_mat =5  
 );  
END;  
  
SELECT *inscritos*() as ins;  
  
SELECT est.nombres, est.apellidos, mat.nombre\_mat  
FROM estudiantes AS est, materias as mat  
where mat.id\_mat=*inscritos*();  
  
create or replace function *sum\_edad*(edad integer, genero varchar(20))  
returns integer  
begin  
 return (  
 select sum(est.edad)  
 from estudiantes as est  
 where est.sexo=genero and est.edad >edad  
 );  
end;  
  
SELECT est.nombres,est.apellidos  
 FROM estudiantes AS est  
 where *sum\_edad*( 18, 'masculino') % 2 = 0;  
  
create or replace function *compara*(nombre varchar(50), apellido varchar(50))  
returns integer  
begin  
 return (  
 select (est.nombres)  
 from estudiantes as est  
 where est.nombres='Miguel' and est.apellidos='Gonzales Veliz'  
 );  
end;  
  
SELECT est.\*  
 FROM estudiantes AS est  
 where *compara*( 'Santos', 'Montes Valenzuela');