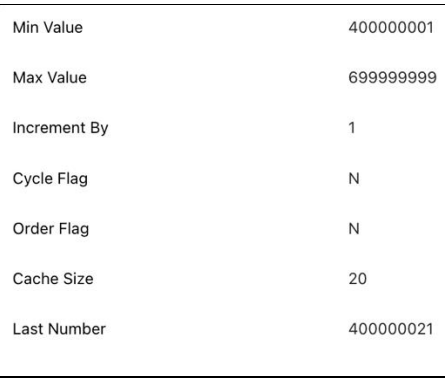
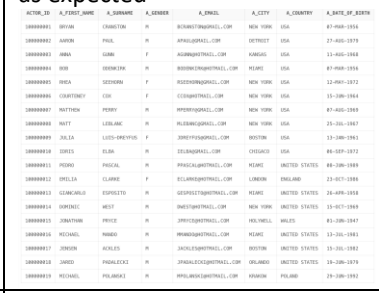
TEST PLAN

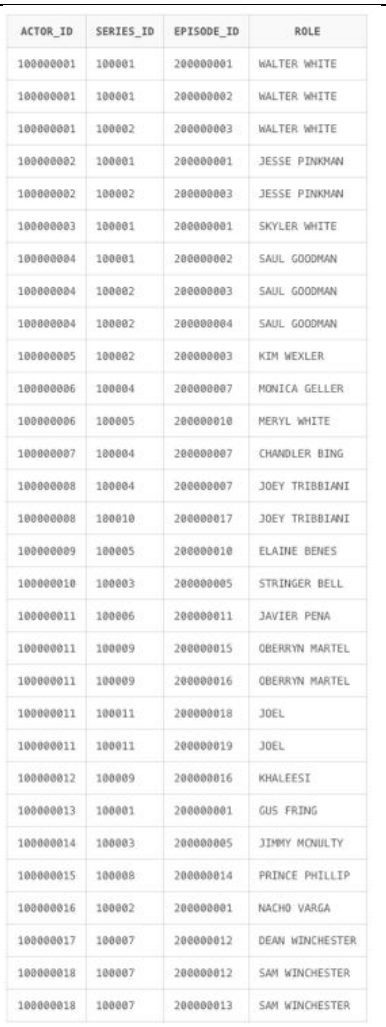
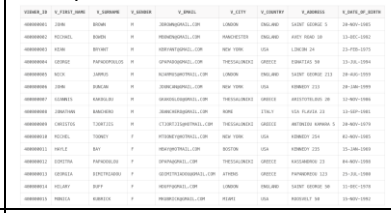
A. create

ID	Test	Element tested	Expected Result	Actual Result and Screenshot
1	CREATE TABLE actors(Actor_id NUMBER(9), A_First_Name VARCHAR2(30), A_Surname VARCHAR2(30), A_Gender CHAR(1), A_Email VARCHAR2(50), A_City VARCHAR2(30), A_Country VARCHAR2(15), A_Date_of_birth DATE); desc actors	creation of a new table named actors with 8 columns	a table for the actors with the appropriate columns	as expected TABLE ACTORS Column / Null? / Type ACTOR_ID – NOT NULL – NUMBER(9,0) A_FIRST_NAME – – VARCHAR2(30) A_SURNAME – – VARCHAR2(30) A_GENDER – – CHAR(1) A_EMAIL – – VARCHAR2(50) A_CITY – – VARCHAR2(30) A_COUNTRY – – VARCHAR2(15) A_DATE_OF_BIRTH – – DATE
2	CREATE TABLE series(Series_id NUMBER(6), S_Title VARCHAR2(50), Description VARCHAR2(50), Episodes NUMBER(3), Seasons NUMBER(2), Year DATE); desc series	creation of a new table named series with 6 columns	a table for the series with the appropriate columns	as expected TABLE SERIES Column / Null? / Type SERIES_ID – NOT NULL – NUMBER(6,0) S_TITLE – – VARCHAR2(50) DESCRIPTION – – VARCHAR2(50) EPISODES – – NUMBER(3,0) SEASONS – – NUMBER(2,0) YEAR – – DATE
3	CREATE TABLE casting(Actor_id NUMBER(9), Series_id NUMBER(6), Episode_id NUMBER(9), Role VARCHAR2(50));Year DATE); desc casting	creation of a new table named casting with 4 columns	a table for the casting with the appropriate columns	as expected TABLE CASTING Column / Null? / Type ACTOR_ID – NUMBER(9,0) SERIES_ID – NUMBER(6,0) EPISODE_ID – NUMBER(9,0) ROLE – VARCHAR2(50)
4	CREATE TABLE episode(Episode_id NUMBER(9), Series_id NUMBER(6), Season NUMBER(2), E_Title VARCHAR2(50), Length NUMBER(6)); desc episode	creation of a new table named episode with 5 columns	a table for the episode with the appropriate columns	as expected TABLE EPISODE Column / Null? / Type EPISODE_ID – NOT NULL – NUMBER(9,0) SERIES_ID – – NUMBER(6,0) SEASON – – NUMBER(2,0) E_TITLE – – VARCHAR2(50) LENGTH – – NUMBER(6,0)
5	CREATE TABLE viewer(Viewer_id NUMBER(9), V_First_Name VARCHAR2(10), V_Surname VARCHAR2(20), V_Gender CHAR(1), V_Email VARCHAR2(50),	creation of a new table named viewer with 9 columns	a table for the viewers with the appropriate columns	as expected

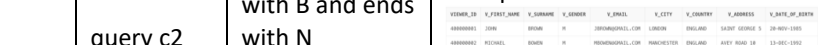
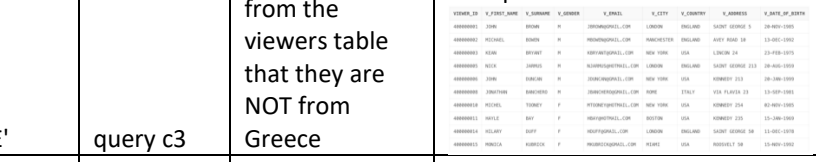
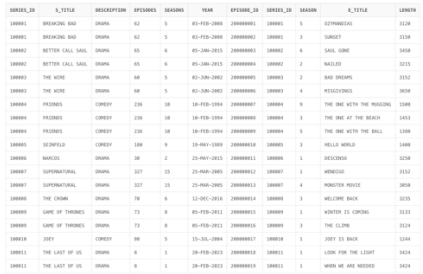
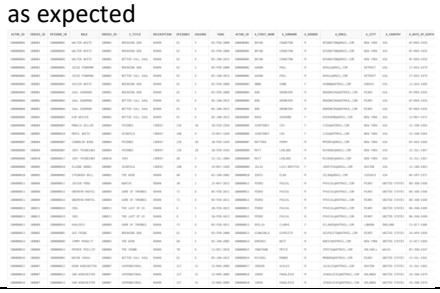
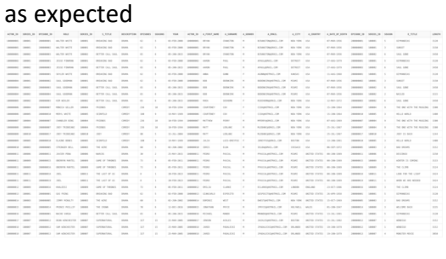
	V_City VARCHAR2(30), V_Country VARCHAR2(35), V_Address VARCHAR2(35), V_Date_of_birth DATE); desc viewer			TABLE VIEWER Column / Null? / Type VIEWER_ID – NOT NULL – NUMBER(9,0) V_FIRST_NAME – VARCHAR2(10) V_SURNAME – VARCHAR2(20) V_GENDER – CHAR(1) V_EMAIL – VARCHAR2(50) V_CITY – VARCHAR2(30) V_COUNTRY – VARCHAR2(15) V_ADDRESS – VARCHAR2(35) V_DATE_OF_BIRTH – DATE
6	CREATE TABLE viewership(Viewer_id NUMBER(9), Episode_id NUMBER(9), Series_id NUMBER(6), Date_watched DATE); desc viewership	creation of a new table named viewership with 4 columns	a table for the viewerships with the appropriate columns	as expected TABLE VIEWERSHIP Column / Null? / Type VIEWER_ID – NUMBER(9,0) EPISODE_ID – NUMBER(9,0) SERIES_ID – NUMBER(6,0) DATE_WATCHED – DATE
7	CREATE SEQUENCE seq_actors START WITH 100000001 INCREMENT BY 1 MINVALUE 100000001 MAXVALUE 199999999 NOCYCLE;	creation of a sequence for actors ID	a sequence of numbers between the MINVALUE and the MAXVALUE chosen and increasing by step 1	as expected Min Value 100000001 Max Value 199999999 Increment By 1 Cycle Flag N Order Flag N Cache Size 20 Last Number 100000021
8	CREATE SEQUENCE seq_series START WITH 100001 INCREMENT BY 1 MINVALUE 100001 MAXVALUE 199999 NOCYCLE;	creation of a sequence for series ID	a sequence of numbers between the MINVALUE and the MAXVALUE chosen and increasing by step 1	as expected Min Value 100001 Max Value 199999 Increment By 1 Cycle Flag N Order Flag N Cache Size 20 Last Number 100021
9	CREATE SEQUENCE seq_episodes START WITH 200000001 INCREMENT BY 1 MINVALUE 200000001 MAXVALUE 399999999 NOCYCLE;	creation of a sequence for episodes ID	a sequence of numbers between the MINVALUE and the MAXVALUE chosen and increasing by step 1	as expected Min Value 200000001 Max Value 399999999 Increment By 1 Cycle Flag N Order Flag N Cache Size 20 Last Number 200000021
10	CREATE SEQUENCE seq_viewers START WITH 400000001 INCREMENT BY 1 MINVALUE 400000001 MAXVALUE 699999999 NOCYCLE;	creation of a sequence for viewers ID	a sequence of numbers between the MINVALUE and the MAXVALUE chosen and	as expected

			increasing by step 1	

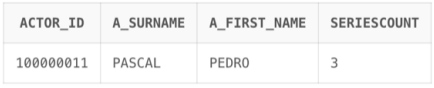
ID	Test	Element tested	Expected Result	Actual Result and Screenshot
1	SELECT * FROM actors;	query a1	all the 8 columns with all the data from actors' table	as expected 
2	SELECT * FROM series;	query a2	all the 6 columns with all the data from series' table	as expected 
3	SELECT * FROM casting;	query a3	all the 4 columns with all the data from casting's table	as expected

				 ACTOR_ID | SERIES_ID | EPISODE_ID | ROLE 100000001 | 100001 | 200000001 | WALTER WHITE 100000001 | 100001 | 200000002 | WALTER WHITE 100000001 | 100002 | 200000003 | WALTER WHITE 100000002 | 100001 | 200000001 | JESSE PINKMAN 100000002 | 100002 | 200000003 | JESSE PINKMAN 100000003 | 100001 | 200000001 | SKYLER WHITE 100000004 | 100001 | 200000002 | SAUL GOODMAN 100000004 | 100002 | 200000003 | SAUL GOODMAN 100000004 | 100002 | 200000004 | SAUL GOODMAN 100000005 | 100002 | 200000003 | KIM WEXLER 100000006 | 100004 | 200000007 | MONICA GELLER 100000006 | 100005 | 200000010 | MERYL WHITE 100000007 | 100004 | 200000007 | CHANDLER BING 100000008 | 100004 | 200000007 | JOEY TRIBBIANI 100000008 | 100010 | 200000017 | JOEY TRIBBIANI 100000009 | 100005 | 200000010 | ELAINE BENES 100000010 | 100003 | 200000005 | STRINGER BELL 100000011 | 100006 | 200000011 | JAVIER PENA 100000011 | 100009 | 200000015 | OBERRYN MARTEL 100000011 | 100009 | 200000016 | OBERRYN MARTEL 100000011 | 100011 | 200000018 | JOEL 100000011 | 100011 | 200000019 | JOEL 100000012 | 100009 | 200000016 | KHALEESI 100000013 | 100001 | 200000001 | GUS FRING 100000014 | 100003 | 200000005 | JIMMY MCNULTY 100000015 | 100008 | 200000014 | PRINCE PHILLIP 100000016 | 100002 | 200000001 | NACHO VARGA 100000017 | 100007 | 200000012 | DEAN WINCHESTER 100000018 | 100007 | 200000012 | SAM WINCHESTER 100000018 | 100007 | 200000013 | SAM WINCHESTER
4	SELECT * FROM episode;	query a4	all the 5 columns with all the data from episodes' table	as expected  EPISODE_ID | SERIES_ID | SEASON | E_TITLE | LENGTH 200000001 | 100001 | 5 | OZYMANDIAS | 3120 200000002 | 100001 | 3 | SUNSET | 3150 200000003 | 100002 | 6 | SAUL GONE | 3450 200000004 | 100002 | 2 | NAILED | 3215 200000005 | 100003 | 2 | BAD DREAMS | 3152 200000006 | 100003 | 4 | MISGIVINGS | 3650 200000007 | 100004 | 9 | THE ONE WITH THE MUGGING | 1500 200000008 | 100004 | 3 | THE ONE AT THE BEACH | 1453 200000009 | 100004 | 5 | THE ONE WITH THE BALL | 1380 200000010 | 100005 | 3 | HELLO WORLD | 1400 200000011 | 100006 | 1 | DESCENSO | 3250 200000012 | 100007 | 1 | WENDIGO | 3152 200000013 | 100007 | 4 | MONSTER MOVIE | 3050 200000014 | 100008 | 3 | WELCOME BACK | 3235 200000015 | 100009 | 1 | WINTER IS COMING | 3133 200000016 | 100009 | 3 | THE CLIMB | 3124 200000017 | 100010 | 1 | JOEY IS BACK | 1244 200000018 | 100011 | 1 | LOOK FOR THE LIGHT | 3424 200000019 | 100011 | 1 | WHEN WE ARE NEEDED | 3424
5	SELECT * FROM viewer;	query a5	all the 9 columns with all the data from viewers' table	as expected 
6	SELECT * FROM viewership;	query a6	all the 4 columns with all the data from viewerships' table	as expected

				VIEWER_ID | EPISODE_ID | SERIES_ID | DATE_WATCHED 400000001 | 200000001 | 100001 | 15-JAN-2020 400000002 | 200000002 | 100001 | 12-MAY-2009 400000002 | 200000003 | 100002 | 05-MAR-2022 400000003 | 200000005 | 100003 | 25-DEC-2015 400000003 | 200000006 | 100003 | 17-AUG-2019 400000004 | 200000007 | 100004 | 03-JAN-2020 400000004 | 200000010 | 100005 | 09-JUN-2018 400000004 | 200000011 | 100006 | 10-JUL-2017 400000005 | 200000012 | 100007 | 01-OCT-2023 400000006 | 200000013 | 100007 | 09-SEP-2020 400000006 | 200000014 | 100008 | 05-FEB-2019 400000007 | 200000015 | 100009 | 06-JAN-2016 400000008 | 200000017 | 100010 | 02-FEB-2020 400000009 | 200000018 | 100011 | 19-APR-2018 400000010 | 200000019 | 100011 | 18-NOV-2014 400000011 | 200000019 | 100011 | 17-MAR-2019 400000011 | 200000001 | 100001 | 25-DEC-2022 400000012 | 200000010 | 100005 | 13-AUG-2017 400000013 | 200000004 | 100002 | 08-JAN-2022 400000014 | 200000008 | 100004 | 20-DEC-2023 400000015 | 200000019 | 100011 | 20-DEC-2023
7	SELECT Length, E_Title, Series_id FROM episode ORDER BY Length DESC;	query b	A projection with 3 columns of the episode table(length, episode's title and series id) sorted in reverse alphabetical order	as expected LENGTH | E_TITLE | SERIES_ID 3650 | MISGIVINGS | 100003 3450 | SAUL GONE | 100002 3424 | LOOK FOR THE LIGHT | 100011 3424 | WHEN WE ARE NEEDED | 100011 3250 | DESCENSO | 100006 3235 | WELCOME BACK | 100008 3215 | NAILED | 100002 3152 | WENDIGO | 100007 3152 | BAD DREAMS | 100003 3150 | SUNSET | 100001 3133 | WINTER IS COMING | 100009 3124 | THE CLIMB | 100009 3120 | OZYMANDIAS | 100001 3050 | MONSTER MOVIE | 100007 1500 | THE ONE WITH THE MUGGING | 100004 1453 | THE ONE AT THE BEACH | 100004 1400 | HELLO WORLD | 100005 1380 | THE ONE WITH THE BALL | 100004 1244 | JOEY IS BACK | 100010
8	SELECT A_First_Name, A_Surname, A_Country FROM actors WHERE A_Country = 'USA';	query c1	A table with 3 columns first name, surname and county only of the actors that live in USA	as expected A_FIRST_NAME | A_SURNAME | A_COUNTRY BRYAN | CRANSTON | USA AARON | PAUL | USA ANNA | GUNN | USA BOB | ODENKIRK | USA RHEA | SEEHORN | USA COURTENEY | COX | USA MATTHEW | PERRY | USA MATT | LEBLANC | USA JULIA | LUIS-DREYFUS | USA IDRIS | ELBA | USA

9	SELECT * FROM viewer WHERE V_Surname LIKE 'B%N';	query c2	A table with all the elements from the viewers table that their Surname starts with B and ends with N	as expected 
10	SELECT * FROM viewer WHERE V_Country NOT IN 'GREECE'	query c3	A table with all the elements from the viewers table that they are NOT from Greece	as expected 
11	SELECT viewer.V_First_Name, viewer.V_Surname, viewer.Viewer_id, viewership.Date_watched FROM viewer INNER JOIN viewership on viewer.Viewer_id=viewership.Viewer_id WHERE Date_watched BETWEEN '01-JAN-2019' AND '31-DEC-2021';	query c4	A table with 3 columns, first name, surname, viewer id from the viewers table and date watched from the viewership's table where a viewer watched an episode from 01 of January of 2019 until 31 of December of 2021	as expected V_FIRST_NAME | V_SURNAME | VIEWER_ID | DATE_WATCHED JOHN | BROWN | 400000001 | 15-JAN-2020 KEAN | BRYANT | 400000003 | 17-AUG-2019 GEORGE | PAPADOPOULOS | 400000004 | 03-JAN-2020 JOHN | DUNCAN | 400000006 | 09-SEP-2020 JOHN | DUNCAN | 400000006 | 05-FEB-2019 JONATHAN | BANCHERO | 400000008 | 02-FEB-2020 HAYLE | BAY | 400000011 | 17-MAR-2019
12	SELECT * FROM series INNER JOIN episode ON series.Series_id=episode.Series_id;	query d1	Join of 2 tables (6+5=11 columns) series and episode	as expected 
13	SELECT * FROM casting INNER JOIN series ON casting.Series_id=series.Series_id INNER JOIN actors ON casting.Actor_id=actors.Actor_id;	query d2	Join of 3 tables series, actors and casting	as expected 
14	SELECT * FROM casting INNER JOIN series ON casting.Series_id=series.Series_id INNER JOIN actors ON casting.Actor_id=actors.Actor_id INNER JOIN episode ON casting.Episode_id=episode.Episode_id;	query d3	Join of 4 tables series, actors,episode and casting	as expected 

15	SELECT a_first_name,a_surname FROM actors where a_surname LIKE 'C%E';	query e1	Shows the first name and the surname of the actors that their surname starts with C and ends with E. (table: 1 line x 2 columns)	as expected A_FIRST_NAME: EMILIA A_SURNAME: CLARKE
16	SELECT v_country, COUNT(v_country) FROM viewer GROUP BY v_country;	query e2	It counts the total number of viewers per country and groups the result. (table: 4 lines x 2 columns)	as expected V_COUNTRY / COUNT(V_COUNTRY): ITALY 1 ENGLAND 4 USA 5 GREECE 5
17	SELECT episode.Episode_id, episode.E_Title, COUNT(viewership.Viewer_id) AS ViewCount FROM episode JOIN viewership ON episode.Episode_id = viewership.Episode_id GROUP BY episode.Episode_id, episode.E_Title ORDER BY ViewCount DESC FETCH FIRST 1 ROW ONLY;	query e3	It counts the total number of viewers for each episode, then puts the episodes in descending order and keeps the 1st line , which is the most popular episode. (table: 1 line x 3 columns)	as expected EPISODE_ID: 200000019 E_TITLE: WHEN WE ARE NEEDED VIEWCOUNT: 3
18	SELECT series.Series_id, series.S_Title, COUNT(viewership.Viewer_id) AS ViewCount FROM series JOIN viewership ON series.Series_id = viewership.Series_id GROUP BY series.Series_id, series.S_Title ORDER BY ViewCount DESC FETCH FIRST 1 ROW ONLY;	query e4	It counts the total number of viewers for each series, then puts the series in descending order and keeps the 1st line , which is the most popular series. (table: 1 line x 3 columns)	as expected SERIES_ID: 100011 S_TITLE: THE LAST OF US VIEWCOUNT: 4

19	SELECT actors.Actor_id, actors.A_Surname, actors.A_First_Name, series.S_Title FROM actors JOIN casting ON actors.Actor_id = casting.Actor_id JOIN series ON casting.Series_id = series.Series_id ORDER BY actors.A_Surname ASC, actors.A_First_Name ASC, actors.Actor_id ASC;	query e5	It shows all the actors id, surname and first name and all the series that they participated in ascending order.	as expected 
20	SELECT actors.Actor_id, actors.A_Surname, actors.A_First_Name, COUNT(DISTINCT casting.Series_id) AS SeriesCount FROM actors JOIN casting ON actors.Actor_id = casting.Actor_id GROUP BY actors.Actor_id, actors.A_Surname, actors.A_First_Name ORDER BY SeriesCount DESC FETCH FIRST 1 ROW ONLY;	query e6	It counts the number of series that every actor participated and then puts them in a descending order and keeps the first one , which Is the actor that participated in the most series. (table: 1 line x 4 columns)	as expected 
21	SELECT actors.Actor_id, actors.A_Surname, actors.A_First_Name, COUNT(DISTINCT casting.Episode_id) AS EpisodeCount FROM actors JOIN casting ON actors.Actor_id = casting.Actor_id GROUP BY actors.Actor_id, actors.A_Surname, actors.A_First_Name ORDER BY EpisodeCount DESC FETCH FIRST 1 ROW ONLY;	query e7	It counts the number of episodes that every actor participated and then puts them in a descending order and keeps the first one , which is the actor that participated in the most episodes.	as expected 

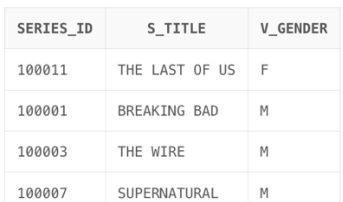
ACTOR_ID	A_SURNAME	A_FIRST_NAME	S_TITLE
100000017	ACKLES	JENSEN	SUPERNATURAL
100000012	CLARKE	EMILIA	GAME OF THRONES
100000006	COX	COURTENEY	FRIENDS
100000006	COX	COURTENEY	SEINFELD
100000001	CRANSTON	BRYAN	BREAKING BAD
100000001	CRANSTON	BRYAN	BREAKING BAD
100000001	CRANSTON	BRYAN	BETTER CALL SAUL
100000010	ELBA	IDRIS	THE WIRE
100000013	ESPOSITO	GIANCARLO	BREAKING BAD
100000003	GUNN	ANNA	BREAKING BAD
100000008	LEBLANC	MATT	JOEY
100000008	LEBLANC	MATT	FRIENDS
100000009	LUIS-DREYFUS	JULIA	SEINFELD
100000016	MANDO	MICHAEL	BETTER CALL SAUL
100000004	ODENKIRK	BOB	BREAKING BAD
100000004	ODENKIRK	BOB	BETTER CALL SAUL
100000004	ODENKIRK	BOB	BETTER CALL SAUL
100000018	PADALECKI	JARED	SUPERNATURAL
100000018	PADALECKI	JARED	SUPERNATURAL
100000011	PASCAL	PEDRO	GAME OF THRONES
100000011	PASCAL	PEDRO	THE LAST OF US
100000011	PASCAL	PEDRO	THE LAST OF US
100000011	PASCAL	PEDRO	NARCOS
100000011	PASCAL	PEDRO	GAME OF THRONES
100000002	PAUL	AARON	BETTER CALL SAUL
100000002	PAUL	AARON	BREAKING BAD
100000007	PERRY	MATTHEW	FRIENDS
100000015	PRYCE	JONATHAN	THE CROWN
100000005	SEEHORN	RHEA	BETTER CALL SAUL
100000014	WEST	DOMINIC	THE WIRE

ACTOR_ID	A_SURNAME	A_FIRST_NAME	SERIESCOUNT
100000011	PASCAL	PEDRO	3

ACTOR_ID	A_SURNAME	A_FIRST_NAME	EPISODECOUNT
100000011	PASCAL	PEDRO	5

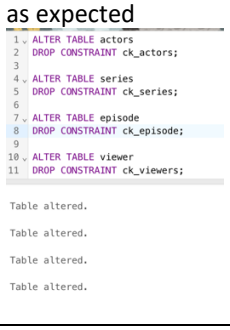
			(table: 1 line x 4 columns)	
22	SELECT S_Title FROM series ORDER BY Seasons DESC FETCH FIRST 1 ROW ONLY;	query e8	It puts the series in descending order regarding their number of season and shows the 1st title of the series which is the one with the most seasons. (table: 1 line x 1 column)	as expected S_TITLE SUPERNATURAL
23	SELECT S_Title FROM series ORDER BY Episodes DESC FETCH FIRST 1 ROW ONLY;	query e9	It puts the series in descending order regarding their number of episodes and shows the 1st title of the series which is the one with the most episodes in total. (table: 1 line x 1 column)	as expected S_TITLE SUPERNATURAL
24	SELECT COUNT(*) AS SeriesCount FROM series WHERE EXTRACT(Year FROM Year) BETWEEN 2000 AND 2009;	query e10	It counts the number of series produced between 2000 and 2009 and saves this number as SeriesCount. (table: 1 line x 1 column)	as expected SERIESCOUNT 4
25	SELECT Actor_id, A_First_Name, A_Surname, A_Date_of_birth FROM actors WHERE EXTRACT(MONTH FROM A_Date_of_birth) = 7;	query e11	It shows the actors ID, firstname, surname and date of birth of the actors that were born in July , regardless the year .	as expected ACTOR_ID | A_FIRST_NAME | A_SURNAME | A_DATE_OF_BIRTH 100000008 | MATT | LEBLANC | 25-JUL-1967 100000016 | MICHAEL | MANDO | 13-JUL-1981 100000017 | JENSEN | ACKLES | 15-JUL-1982

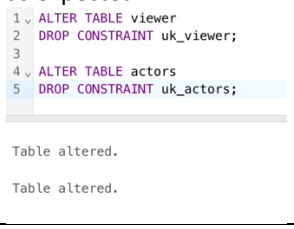
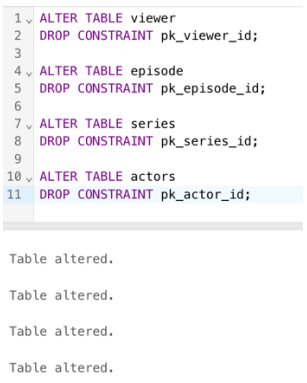
26	SELECT COUNT(*) AS ActorCount FROM actors LEFT JOIN casting ON actors.Actor_id = casting.Actor_id WHERE casting.Actor_id IS NULL;	query e12	It shows the number of actors that haven't participated in a series in the database, hence casting.Actor_id IS NULL. (table: 1 line x 1 column)	as expected ACTORCOUNT 1
27	SELECT V_First_Name, V_Surname, V_Date_of_birth FROM viewer WHERE V_Gender = 'F' ORDER BY V_Date_of_birth ASC FETCH FIRST 1 ROW ONLY;	query e13	It puts in descending order all the female viewers regarding their date of birth and then shows the 1st one which is the oldest female viewer. (table: 1 line x 3 columns)	as expected V_FIRST_NAME V_SURNAME V_DATE_OF_BIRTH HAYLE BAY 15-JAN-1969
28	SELECT Viewer_id, V_First_Name, V_Surname, V_Address FROM viewer WHERE UPPER(V_Address) LIKE '%SAINT%';	query e14	It shows the viewer ID, the firstname, the surname and the address of the viewers that leave in address with the word saint in it.	as expected VIEWER_ID V_FIRST_NAME V_SURNAME V_ADDRESS 400000001 JOHN BROWN SAINT GEORGE 5 400000005 NICK JARMUS SAINT GEORGE 213 400000014 HILARY DUFF SAINT GEORGE 50
29	SELECT series.S_Title, AVG(episode.Length) AS AvgEpisodeLength FROM series JOIN episode ON series.Series_id = episode.Series_id GROUP BY series.Series_id, series.S_Title ORDER BY AvgEpisodeLength DESC;	query e15	It calculates the average of the episode's length for each series and the puts them in descending order.	as expected S_TITLE AVGEPISODELENGTH THE LAST OF US 3424 THE WIRE 3401 BETTER CALL SAUL 3332.5 NARCOS 3250 THE CROWN 3235 BREAKING BAD 3135 GAME OF THRONES 3128.5 SUPERNATURAL 3101 FRIENDS 1444.3333333333333333333333333333333333 SEINFELD 1400 JOEY 1244
30	SELECT viewer.Viewer_id,	query e16	It counts the number of	as expected

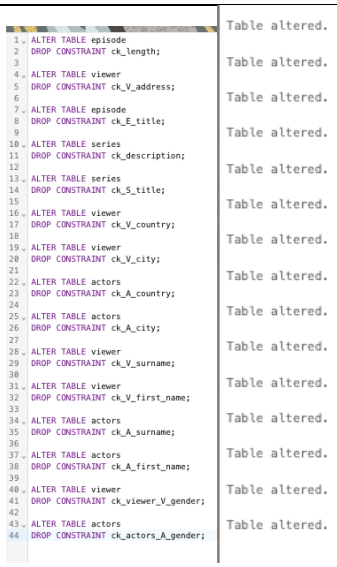
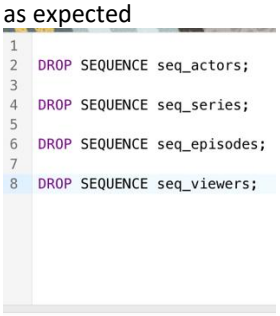
	viewer.V_First_Name, viewer.V_Surname, COUNT(DISTINCT viewership.Episode_id) AS ECount FROM viewer JOIN viewership ON viewer.Viewer_id = viewership.Viewer_id GROUP BY viewer.Viewer_id, viewer.V_First_Name, viewer.V_Surname ORDER BY ECount DESC FETCH FIRST 1 ROW ONLY;		episodes that each viewer has watched, then puts the viewers in descending order and keeps the 1st one which is the viewer has watched the most episodes. (table: 1 line x 4 columns)	 VIEWER_ID V_FIRST_NAME V_SURNAME ECOUNT 400000004 GEORGE PAPADOPOULOS 3
31	SELECT viewer.Viewer_id, viewer.V_First_Name, viewer.V_Surname, COUNT(DISTINCT viewership.Episode_id) AS ECount FROM viewer JOIN viewership ON viewer.Viewer_id = viewership.Viewer_id WHERE UPPER(viewer.V_City) = 'THESSALONIKI' GROUP BY viewer.Viewer_id, viewer.V_First_Name, viewer.V_Surname ORDER BY ECount DESC FETCH FIRST 1 ROW ONLY;	query e17	It counts the number of episodes that each viewer placed in Thessaloniki has watched, then puts the viewers in descending order and keeps the 1st one which is the viewer has watched the most episodes in Thessaloniki. (table: 1 line x 4 columns)	as expected  VIEWER_ID V_FIRST_NAME V_SURNAME ECOUNT 400000004 GEORGE PAPADOPOULOS 3
32	WITH RankedSeries AS (SELECT viewer.V_Gender, series.Series_id, series.S_Title, RANK() OVER (PARTITION BY viewer.V_Gender ORDER BY COUNT(viewership.Episode_id) DESC) AS SeriesRank FROM viewership JOIN series ON viewership.Series_id = series.Series_id JOIN viewer ON viewership.Viewer_id = viewer.Viewer_id GROUP BY viewer.V_Gender, series.Series_id, series.S_Title) SELECT Series_id,	query e18	First we calculate the number of episodes watched based on the each gender separately and then we display each series' title and the gender that watched it the most.	as expected  SERIES_ID S_TITLE V_GENDER 100011 THE LAST OF US F 100001 BREAKING BAD M 100003 THE WIRE M 100007 SUPERNATURAL M

	S_Title, V_Gender FROM RankedSeries WHERE SeriesRank = 1;			
33	SELECT Viewer_id, V_First_Name, V_Surname, SUBSTR(UPPER(V_First_Name), 1, 4) || SUBSTR(UPPER(V_Surname), 1, 4) AS Username FROM viewer;	query e19	We created a new column called username where it's a string is formed by 8 letters, the first 4 are the the first 4 letters of the first name and the other 4 are the the first 4 letters of the surname. (table with 4 columns)	as expected VIEWER_ID | V_FIRST_NAME | V_SURNAME | USERNAME 400000001 | JOHN | BROWN | JOHNBROW 400000002 | MICHAEL | BOWEN | MICHBOWE 400000003 | KEAN | BRYANT | KEANBRYA 400000004 | GEORGE | PAPADOPOULOS | GEORPAPA 400000005 | NICK | JARMUS | NICKJARM 400000006 | JOHN | DUNCAN | JOHNDUNC 400000007 | GIANNIS | KAKOGLOU | GIANKAKO 400000008 | JONATHAN | BANCHERO | JONABANC 400000009 | CHRISTOS | TJORTJIS | CHRITJOR 400000010 | MICHEL | TOONEY | MICHTOON 400000011 | HAYLE | BAY | HAYLBAY 400000012 | DIMITRA | PAPADOULOU | DIMIPAPA 400000013 | GEORGIA | DIMITRIADOU | GEORDIMI 400000014 | HILARY | DUFF | HILADUFF 400000015 | MONICA | KUBRICK | MONIKUBR
34	SELECT EXTRACT(YEAR FROM series.Year) AS ProductionYear, SUM(episode.Length) AS TotalSeriesLength FROM series JOIN episode ON series.Series_id = episode.Series_id GROUP BY EXTRACT(YEAR FROM series.Year) ORDER BY ProductionYear;	query e20	It calculates the sum of all the episodes produced each year and it shows the total length of all the series per year. (table with 2 columns)	as expected PRODUCTIONYEAR | TOTALSERIESLENGTH 1989 | 1400 1994 | 4333 2002 | 6802 2004 | 1244 2005 | 6202 2008 | 6270 2011 | 6257 2015 | 9915 2016 | 3235 2023 | 6848

DROP TABLES

ID	Test	Element tested	Expected Result	Actual Result and Screenshot
1	ALTER TABLE actors DROP CONSTRAINT ck_actors; ALTER TABLE series DROP CONSTRAINT ck_series; ALTER TABLE episode DROP CONSTRAINT ck_episode; ALTER TABLE viewer DROP CONSTRAINT ck_viewers;	DROP CONSTRAINTS	DROP CHECK CONSTRAINTS	as expected 

2	ALTER TABLE casting DROP CONSTRAINT fk_actor_id; ALTER TABLE casting DROP CONSTRAINT fk_series_id; ALTER TABLE casting DROP CONSTRAINT fk_episode_id; ALTER TABLE episode DROP CONSTRAINT fk_episode_series_id; ALTER TABLE viewership DROP CONSTRAINT fk_viewer_id; ALTER TABLE viewership DROP CONSTRAINT fk_viewership_series_id; ALTER TABLE viewership DROP CONSTRAINT fk_viewership_episode_id;	DROP CONSTRAINTS	DROP FK CONSTRAINTS	as expected 
3	ALTER TABLE viewer DROP CONSTRAINT uk_viewer; ALTER TABLE actors DROP CONSTRAINT uk_actors;	DROP CONSTRAINTS	DROP UK CONSTRAINTS	as expected 
4	ALTER TABLE viewer DROP CONSTRAINT pk_viewer_id; ALTER TABLE episode DROP CONSTRAINT pk_episode_id; ALTER TABLE series DROP CONSTRAINT pk_series_id; ALTER TABLE actors DROP CONSTRAINT pk_actor_id;	DROP CONSTRAINTS	DROP PK CONSTRAINTS	as expected 
5	ALTER TABLE episode DROP CONSTRAINT ck_length; ALTER TABLE viewer DROP CONSTRAINT ck_V_address; ALTER TABLE episode DROP CONSTRAINT ck_E_title; ALTER TABLE series DROP CONSTRAINT ck_description; ALTER TABLE series DROP CONSTRAINT ck_S_title;	DROP CONSTRAINTS	DROP CHECK CONSTRAINTS	as expected

	ALTER TABLE viewer DROP CONSTRAINT ck_V_country; ALTER TABLE viewer DROP CONSTRAINT ck_V_city; ALTER TABLE actors DROP CONSTRAINT ck_A_country; ALTER TABLE actors DROP CONSTRAINT ck_A_city; ALTER TABLE viewer DROP CONSTRAINT ck_V_surname; ALTER TABLE viewer DROP CONSTRAINT ck_V_first_name; ALTER TABLE actors DROP CONSTRAINT ck_A_surname; ALTER TABLE actors DROP CONSTRAINT ck_A_first_name; ALTER TABLE viewer DROP CONSTRAINT ck_viewer_V_gender; ALTER TABLE actors DROP CONSTRAINT ck_actors_A_gender;			
6	DROP SEQUENCE seq_actors; DROP SEQUENCE seq_series; DROP SEQUENCE seq_episodes; DROP SEQUENCE seq_viewers;	DROP SEQUENCES	DROP SEQUENCES	as expected 
7	DROP TABLE actors; DROP TABLE series; DROP TABLE casting; DROP TABLE episode; DROP TABLE viewer; DROP TABLE viewership;	DROP TABLES	DROP TABLES	as expected

```sql
DROP TABLE actors;

DROP TABLE series;

DROP TABLE casting;

DROP TABLE episode;

DROP TABLE viewer;

DROP TABLE viewership;
```

```
Table dropped.

Table dropped.

Table dropped.

Table dropped.

Table dropped.

Table dropped.
```