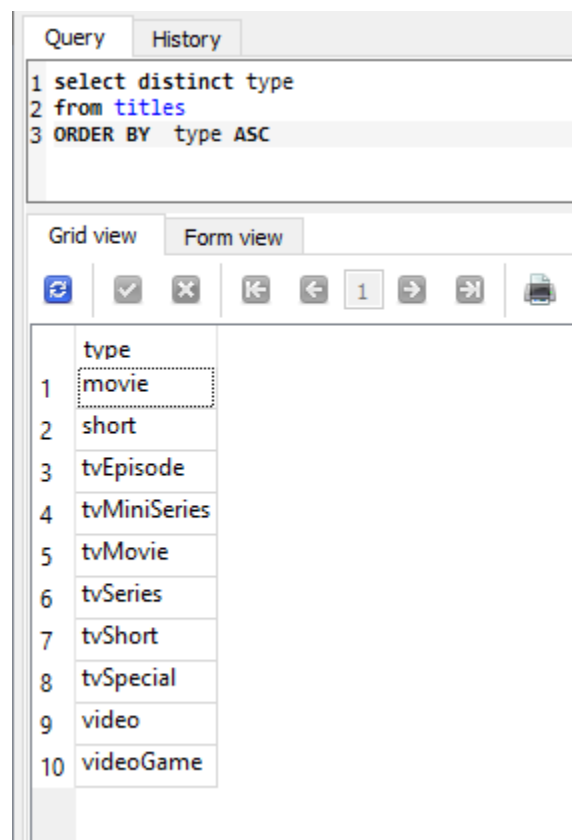


تمرین اول پایگاه داده
علی عسگری
40032223

1 - در این کوئری تمام type ها در جدول titles به شکل غیر تکراری برگشت داده میشوند.

select distinct type
from titles
ORDER BY type ASC

به کمک ASC روند برگشت صعودی خواهد بود اما در این کوئری به ASC هم نیازی نبود و به طور دیفالت به طور صعودی برگشت داده میشود.



The screenshot shows a SQL query editor with a 'Query' tab selected. The query text is:

```
1 select distinct type
2 from titles
3 ORDER BY type ASC
```

Below the query editor, there are tabs for 'Grid view' and 'Form view', with 'Grid view' being the active tab. A toolbar with various icons is visible above the results grid. The results are displayed in a table with 10 rows and 1 column, showing the distinct values of the 'type' column from the 'titles' table, ordered alphabetically (ASC).

	type
1	movie
2	short
3	tvEpisode
4	tvMiniSeries
5	tvMovie
6	tvSeries
7	tvShort
8	tvSpecial
9	video
10	videoGame

2- در این کوئری ، بیشترین مقدار runtime برای هر type پیدا میشود و در خروجی برگردانده میشود.

Query		History
<pre> 1 select primary_title, runtime_minutes, type 2 from titles 3 Group By type 4 having (runtime_minutes) = (select MAX(runtime_minutes)) 5 Order BY type , primary_title ASC 6 </pre>		
Grid view		Form view
		Total rows loaded: 10
primary title	runtime minutes	type
1 Logistics	51420	movie
2 Kuriocity	461	short
3 Téléthon 2012	1800	tvEpisode
4 Kôya no yôjinbô	1755	tvMiniSeries
5 ArtQuench Presents Spirit Art	2112	tvMovie
6 The Sharing Circle	8400	tvSeries
7 Paul McCartney Backstage at Super Bowl XXXIX	60	tvShort
8 Katy Perry Live: Witness World Wide	5760	tvSpecial
9 Midnight Movie Madness: 50 Movie Mega Pack	5135	video
10 Flushy Fish VR: Just Squidding Around	1500	videoGame

با استفاده از ORDER BY ترتیب ها را ابتدا بر اساس نوع و عنوان اصلی قرار میدهیم. (همزمان دو اولویت ترتیب را اعمال کردیم).

Query		History
<pre> 1 select primary_title, runtime_minutes, type 2 from titles 3 Group By type 4 having (runtime_minutes) = (select MAX(runtime_minutes)) 5 order BY type 6 </pre>		
Grid view		Form view
		Total rows loaded: 10
primary title	runtime minutes	type
1 ArtQuench Presents Spirit Art	2112	tvMovie
2 Flushy Fish VR: Just Squidding Around	1500	videoGame
3 Katy Perry Live: Witness World Wide	5760	tvSpecial
4 Kuriocity	461	short
5 Kôya no yôjinbô	1755	tvMiniSeries
6 Logistics	51420	movie
7 Midnight Movie Madness: 50 Movie Mega Pack	5135	video
8 Paul McCartney Backstage at Super Bowl XXXIX	60	tvShort
9 The Sharing Circle	8400	tvSeries
10 Téléthon 2012	1800	tvEpisode

Query		History
<pre> 1 select primary_title, runtime_minutes, type 2 from titles 3 Group By type 4 having (runtime_minutes) = (select MAX(runtime_minutes)) 5 order BY type 6 </pre>		
Grid view		Form view
		Total rows loaded: 10
primary title	runtime minutes	type
1 Logistics	51420	movie
2 Kuriocity	461	short
3 Téléthon 2012	1800	tvEpisode
4 Kôya no yôjinbô	1755	tvMiniSeries
5 ArtQuench Presents Spirit Art	2112	tvMovie
6 The Sharing Circle	8400	tvSeries
7 Paul McCartney Backstage at Super Bowl XXXIX	60	tvShort
8 Katy Perry Live: Witness World Wide	5760	tvSpecial
9 Midnight Movie Madness: 50 Movie Mega Pack	5135	video
10 Flushy Fish VR: Just Squidding Around	1500	videoGame

در دو حالت بالا به طور جداگانه ترتیب type و primary title را اعمال کردیم.

در این کوئری به کمک Group By به مقایسه تمام title هایی که type یکسان دارند می پردازیم و به کمک MAX و HAVING ماکزیمم ران تایم را بدست می آوریم.

3- به کمک **group by** دسته بندی برای هر تایپ را بدست می آوریم ، و در هر دسته به کمک **count** به شمارش تایتل ها می پردازیم و به کمک **order by ASC** بر اساس تعداد شمرده شده صعودی برمیگردانیم.

Query History	
<pre> 1 select type ,count(primary_title) 2 from titles 3 group by type 4 order by count(primary_title) </pre>	
Grid view Form view	
<div> 1 </div> Total rows loaded: 10	
type	count(primary title)
1 tvShort	4075
2 videoGame	9044
3 tvSpecial	9107
4 tvMiniSeries	10291
5 tvMovie	45431
6 tvSeries	63631
7 video	90069
8 movie	197957
9 short	262038
10 tvEpisode	1603076

4- در این کوئری خروجی به شکل نزولی به کمک **ORDER BY count(primary_title) DESC** برگردانده میشود. دسته بندی بر اساس دهه ها به کمک **group by decade** انجام میشود.

به کمک **floor((((premiered) / 10) * 10)|| 's' as decade** توانستیم یک ستون تحت عنوان دهه به خروجی اضافه کنیم ، در واقع نتیجه محاسبات بالا بر روی تاریخ هر فیلد این است که دهه را برای آن فیلد بدست می آورد.

Query History	
<pre> 1 select floor((((premiered) / 10) * 10) 's' as decade ,count(primary_title) as num_titles 2 from titles 3 WHERE decade IS NOT NULL 4 group by decade 5 ORDER BY count(primary_title) DESC 6 </pre>	

<pre> 1 select floor(((premiered) / 10) * 10) 's' as decade ,count(primary_title) as num_titles 2 from titles 3 WHERE decade IS NOT NULL 4 group by decade 5 ORDER BY count(primary_title) DESC </pre>		
Grid view Form view		
<div> 1 </div> Total rows loaded: 16		
	decade	num_titles
1	2010s	1050732
2	2000s	494639
3	1990s	211453
4	1980s	119258
5	1970s	99707
6	1960s	75237
7	1950s	39554
8	1910s	26596
9	1920s	13153
10	1930s	11492
11	1940s	10011
12	1900s	9586
13	2020s	2492
14	1890s	2286
15	1880s	22
16	1870s	1

5- کونری برای این مسئله به شکل زیر است .

تشکل دهه ها به کمک `floor(((premiered) / 10) * 10)|| 's' as decades` میسر میشود .

ستون درصد به کمک خط زیر به دست آمده است.

`ROUND((((count(premiered))*1.0)/((SELECT COUNT(*) FROM titles)*1.0)),6)`

اینجا از `ROUND` برای گرفتن 6 رقم اعشار استفاده شده که در ادامه با ضرب در 100 کردن این عدد درصد مطابق فرمت خواسته شده بدست می آید. ضرب در 1.0 برای تبدیل عدد به `float` استفاده شده.

برای دسته بندی و تشکیل دسته ی هر دهه و ترتیب نزولی بر اساس درصد از دو لاین زیر استفاده شده است.

`group by decades`

`ORDER BY percent DESC`

QueryHistory

```
1 select
2 floor(((premiered) / 10) * 10)|| 's' as decades ,
3 ROUND((((count(premiered))*1.0)/((SELECT COUNT(*) FROM titles)*1.0)),6)*100 as percent
4 from titles
5 WHERE decades IS NOT NULL
6 group by decades
7 ORDER BY percent DESC
```

imdb

QueryHistory

```
1 select
2 floor(((premiered) / 10) * 10)|| 's' as decades ,
3 ROUND((((count(premiered))*1.0)/((SELECT COUNT(*) FROM titles)*1.0)),6)*100 as percent
4 from titles
5 WHERE decades IS NOT NULL
6 group by decades
7 ORDER BY percent DESC
```

Grid viewForm view

✓

✕

⏮

⏪

1

⏩

⏭

🖨

Total rows loaded: 16

	decades	percent
1	2010s	45.7891
2	2000s	21.5555
3	1990s	9.2148
4	1980s	5.1971
5	1970s	4.3451
6	1960s	3.2787
7	1950s	1.7237
8	1910s	1.159
9	1920s	0.5732
10	1930s	0.5008
11	1940s	0.4363
12	1900s	0.4177
13	2020s	0.1086
14	1890s	0.0996
15	1880s	0.001
16	1870s	0

6- در این کوئری با **limit** تعداد سطر های برگشت داده شده را به 10 محدود کردیم.

Query		History
<pre>1 SELECT primary_title,COUNT(*) as num_dubs 2 FROM akas 3 INNER JOIN titles ON titles.title_id = akas.title_id 4 group by akas.title_id 5 ORDER BY COUNT(*) DESC 6 LIMIT 10 7</pre>		
Grid view		Form view
		Total rows loaded: 10
primary title	num dubs	
1 Mutant Virus: Vol. 1	126	
2 The Good, the Bad and the Ugly	73	
3 Star Wars: Episode V - The Empire Strikes Back	71	
4 Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb	68	
5 Raiders of the Lost Ark	62	
6 Star Wars: Episode VII - The Force Awakens	62	
7 The Shawshank Redemption	61	
8 Once Upon a Time in the West	60	
9 Indiana Jones and the Kingdom of the Crystal Skull	60	
10 Airplane!	59	

از جویین بر پایه ی **title_id** که عنصر مشترک **akas** و **titles** بود استفاده کردیم و محتوای برگشتی را بر اساس **title_id** گروه بندی کردیم. ترتیب برگشتی بر اساس تعداد دوبله ها و نزولی به کمک **ORDER BY COUNT(*) DESC** مرتب شده است.

جدول **akas** حاوی نام های هر فیلم به زبان های متفاوت است ، یعنی هر فیلم به تعداد سطر های متعلق به خودش در این جدول دوبله شده . به کمک **title_id** میتوان تشخیص داد که این نام های متفاوت مربوط به یک فیلم میباشند و حال اگر بتوان این ها را بشمارد در واقع تعداد دوبله برای هر فیلم شمرده شده است.

Query History

```

1 select count(name) as count from people
2 where person_id in (select person_id from crew
3 where (category = 'actor' OR category = 'actress') AND title_id in (select title_id from crew
4 where person_id in (select person_id from people
5 where people.name = "Mark Hamill" AND people.born = 1951)
6 ))
7
8
9
10
11

```

Grid view Form view

Total rows loaded: 1

count
1 206

تعداد برگشتی 206 است. نکته مهم در این کوئری انتخاب ویژگی `category = 'actor' OR category = 'actress'` از جدول `crew` میباشد و در ادامه هم باید تاکید کرد که از جدول `people` ویژگی زیر رعایت شود.

people.name = "Mark Hamill" AND people.born = 1951

در واقع ما تعداد نام هایی را به کمک جدول `people` می‌شماریم که `person_id` ایشان در جدول `crew` در حالی برگشت داده شود که ویژگی `category` برای آن سطر یا `actor` باشد یا `actress`، در ادامه این سطر باید در جدول `crew` بر اساس `title_id` مشترک و یکسان و بازگشت آن بر اساس یکسان بودن `person_id` به جدول `people` حاوی نام مارک همیل و بدنیآ آمده در سال 1951 باشد که در نهایت تعداد سطر های برگشتی یا در واقع نام ها را می‌شماریم تا تعداد بازیگر های زن و مردی که با مارک همیل متولد 1951 همبازی بوده اند بدست آید.

میتوان این جست و جو را به بازیگر بودن محدود نکرد ولی بهر حال ما در اینجا این محدودیت را اعمال کرده ایم. در تصویر زیر برخی از نام ها را مشاهده میکنید که حاوی خود مارک همیل نیز هست.

Grid view Form view

Total rows loaded: 206









count	name
1	Tia Carrere
2	Samuel L. Jackson
3	Traci Lords
4	Gary Oldman
5	Bruce Willis
6	Ben Affleck
7	Clancy Brown
8	Robert Englund
9	Carrie Fisher
10	Mark Hamill
11	Christopher Lee
12	Matthew Lillard
13	Ron Perlman
14	Wil Wheaton
15	Demond Acosta

8- در این کوئری بخش مهم نکته زیر است.

name = 'John Wayne' And born=1907 OR name = 'Ward Bond' And born=1903

چند **select** تو در تو داریم و در نهایت یک **group by** که بر اساس **title_id** گروه ها را ایجاد میکند که در واقع در نهایت ما می آیم آن **title_id** ای را برمیگردانیم که در این **group by** دوبار تکرار شده که در واقع با گرفتن **primary_title** این ها از جدول **titles** ، عناوینی را برمیگردانیم که هر دوی این بازیگر در آن پروژه همکاری کرده اند و حضور داشته اند.

در واقع ما در ابتدا تمامی فیلم هایی که هر کدام از این دو ، در آن حضور داشته را به یک لیست جامع اضافه میکنیم.

Grid view	Form view
     1   	Total rows loaded: 125
primary title	
1	The Range Feud
2	Three Girls Lost
3	The Hurricane Express
4	His Private Secretary
5	The Man from Monterey
6	Sagebrush Trail
7	'Neath the Arizona Skies
8	The Lawless Frontier
9	Men of the Night
10	Randy Rides Alone
11	Guard That Girl
12	The New Frontier
13	Rainbow Valley
14	Legion of Terror
15	Sea Spoilers
16	Born to the West
17	Escape by Night
18	Park Avenue Logger

سپس در ادامه با یک **group by** بر روی **title_id** های لیست جامع برگردانده شده ، بر اساس **title_id** گروه بندی میکنیم حال در ادامه با استفاده از **having count(title_id** در این لیست 2 بار تکرار شده ، که به این معناست که یک بار آن را به عنوان فیلمی که بازیگر اول در آن بوده است به لیست جامع اضافه کرده ایم و بار دیگر به عنوان فیلمی که بازیگر دوم در آن بوده است به لیست جامع اضافه کرده ایم. حال این **title_id** هایی که دو بار تکرار شده اند را برمیداریم و به سراغ جدول **titles** میرویم و نام پروژه ها را از آنجا استخراج میکنیم. خروجی نیز بر اساس **primary_title** به شکل صعودی مرتب شده است.

imdb

Query History

```

1 select primary_title from titles
2 where title_id in (select title_id from crew
3 where person_id in (select person_id from people
4 where name = 'John Wayne' And born=1907 OR name = 'Ward Bond' And born=1903)
5 group by title_id
6 having count(title_id) = 2)
7 ORDER BY primary_title

```

Grid view Form view

☒ ☐ ☐ ☐ ☐ ☐ 1 ☐ ☐ ☐

Total rows loaded: 4

	primary title
1	3 Godfathers
2	Homage for The Duke
3	John Wayne: The Unquiet American
4	Operation Pacific

9- در این تمرین یک تریگر مینویسیم که از **AFTER DELETE** استفاده میکند چرا که داستان ما پس از حذف قرار است که رخ دهد ، سپس یک **WHEN** داریم که بررسی میکند یکی از فیلد های سطر حذف شده را که اگر مربوط به اپیزود و سریال مد نظر در سوال بود ، یک اتفاق تحت عنوان **INSERT INTO** رخ دهد و در نهایت مقادیر **OLD** که مقادیر حذف شده هستند بار دیگر وارد جدول دیتای ما شوند.

Query History

```

1 CREATE TRIGGER UndeleteEpisode AFTER DELETE
2 ON episodes
3
4 BEGIN
5 SELECT CASE
6 WHEN OLD.episode_title_id = (SELECT episode_title_id from episodes
7 where show_title_id in (select title_id from titles
8 where primary_title = 'Senifeld' ))
9 THEN RAISE(ABORT,
10 'will add it again.')
11 END;
12 END;
13
14 INSERT INTO episodes
15 (episode_title_id ,show_title_id,season_number, episode_number)
16 VALUES(OLD.episode_title_id,OLD.show_title_id,OLD.season_number,OLD.episode_number);

```