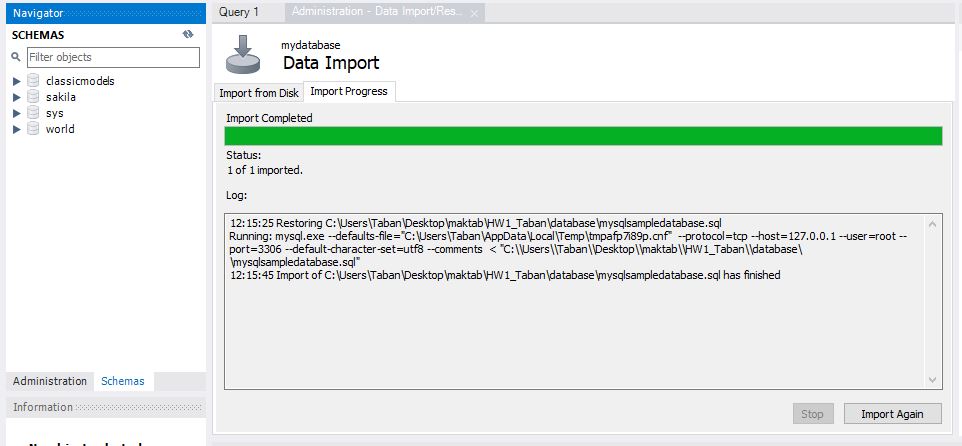
**بخش اول**

در این قسمت، پس از نصب MySQL و MySQL Workbench فایل دیتابیس موجود را Import می­کنیم (مطابق شکل1) و با استفاده از show Query لیست دیتابیس­های موجود را می توان مشاهده کرد.



شکل 1

برای سوال اول از Query نشان داده شده استفاده کردیم، نتایج مربوطه در فایل Questioncsv ذخیره شده است.

SELECT email

FROM classicmodels.employees

WHERE jobTitle = ‘Salees Rep’;

سپس از Query نشان داده شده برای نمایش لیست نام­خانوادگی و کدپستی کسانی که در Spain یا USA هستند، استفاده کردیم و نتایج مربوطه در فایل Question2csv ذخیره شده است.

SELECT contactLastName, postalCode

FROM classicmodels.customers

WHERE country = ‘USA’ OR country = ‘Spain’ ;

برای قسمت سوم، Query نمایش داده شده در کادر زیر را استفاده کردیم و نتایج در فایل Question3csv ذخیره و ضمیمه شده است.

SELECT phone, customerName

FROM classicmodels.customers

WHERE creditLimit < ;

برای قسمت چهارم، با استفاده از INNER JOIN Query نام و قیمت خرید محصولاتی که بیش از 30 عدد سفارش دارند را لیست کرده و نتایج مورد نظر را در فایل Question4csv ذخیره و ضمیمه شده است.

SELECT productName, byPrice

FROM classicmodels.products

INNER JOIN orderdetails ON quantityOrdered > 30;

برای قسمت پنجم نیز با استفاده از Query زیر، تمام ستون­های جدول payment که تاریخ آن به قبل از 2005 مربوط می­شد و مبلغ پرداختی بالای 10 هزار دلار بود را در فایل Question5csv ذخیره کردیم.

SELECT \* FROM classicmodels.product

WHERE (paymentDate BETWEEN ‘0000-00-00’ AND ‘2005-00-00’) AND amount > 10000/00;

5 و 6- دیتابیس­های تکنولوژی NoSQL، اطلاعات را به جای ذخیره­سازی در جداول مرتبط به یکدیگر، آنها را در چندین JSON document یا XML ذخیره می­کنند . باتوجه به انعطاف­پذیری این تکنولوژی (به دلیل دارا بودن ساختار پویا برای ذخیره­ی دیتا) امکان ذخیره­سازی اطلاعات مانند SQL نیز وجود دارد. این دسته از دیتابیس­ها به دلیل انعطاف­پذیری، مقیاس­پذیری بالا و قابلیت پاسخدهی سریع به درخواست­های مدیریت دیتا در صنایع مدرن به کار می­روند. دیتابیس های NoSQL می­توانند تعداد زیادی از کاربران را پشتیبانی کنند، همچنین می­توانند با آپدیت­های متعدد، تغییر نیازها و ویژگی­های جدید خودشان را سازگار کنند و دیتاهای سامان نیافته یا نیمه سامان یافته را نیز هندل می­کنند.

از Relational DBMS بسیار قبل تر از ظهور اینترنت، کلان دیتا، پردازش ابری، موبایل و ... استفاده می شد. و در ابتدا این دسته از دیتابیس­ها برای اجرا بر روی تنها یک سرور ساخته می­شدند. تنها راه برای افزایش ظرفیت این دیتابیس­ها ارتقای آنها برای مقیاس پذیری بالا(به منظور پردازش و ذخیره سازی بهینه­تر , سرعت بالا) بود. از طرفی دیگر، در دیتابیس­های SQL، ذخیره­سازی حجم بالایی از دیتای سازمان نیافته منجر به کاهش سرعت و کارایی دیتابیس می­گردد. بنابراین نیاز بود تا تکنولوژی جدیدی به نام دیتابیس­های NoSQL با هدف ذخیره سازی و کار با دیتای بدون ساختار و حجیم تعریف گردد. به دلیل این که NoSQL باید بتواند انواع مختلف دیتای سازمان نیافته را ذخیره کند در ساختار داخلی آن “Dynamic Schema” به کار رفته است. در مقایسه با SQL، NoSQLقادر به پاسخگویی به کوئری­های پیچیده نمی­باشد و نیز امکان بروز خطای پیش بینی نشده به هنگام ثبت و تغییر دیتا وجود دارد.

انواع دیتابیس­های NoSQL و کاربرد آنها: (همچنین ممکن است این دیتابیسها به صورت تلفیقی به کار روند.)

**دیتابیس­های Key-value NoSQL :** این دسته از دیتابیسها، ساده­ترین و در عین حال پرکاربردترین نوع دیتابیسهای NoSQL هستند. در این دسته، دیتابیس از یک key برای دریافت و ذخیره­سازی value استفاده می­کند.

**دیتابیس­های Document NoSQL :** این دسته از دیتابیس­ها برای ذخیره سازی و کار با document ها با فرمت­های XML, JSON , … به کار می­روند. از این نوع دیتابیسها برای ذخیره­ی دیتای بدون ساختار مشخص با پراکندگی بالا استفاده می­شود.

**دیتابیس­های Wide-column NoSQL :**  این دسته از دیتابیس­ها در ظاهر مانندSQL از جدول استفاده می­کنند اما عملکردشان مانند SQL نمی­باشد. بر خلاف SQL ، در اینجا نوع تعریف و فرمت یک ستون میتواند در هر سطر متفاوت باشد. این دیتابیس­ها دارای انعطاف پذیری بالا در ثبت و کار با کلان دیتاها هستند.

**دیتابیس­های Graph NoSQL:** این دسته، برای ذخیره کردن حجم زیادی از relational data طراحی شده­اند. در شکل هندسی این گونه از دیتابیسها، هر داده به صورت یک رأس و هر لینک به صورت یک یال ترسیم میگردد و از این دیتابیس می توان برای ذخیره­ی انواع معماری­های داده­های شبکه­ای نیز استفاده کرد.

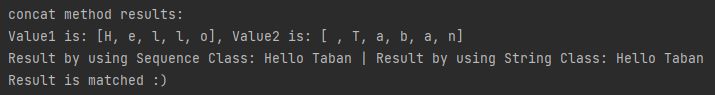
7- از دیتابیس­های in-memory، برای مدیریت سیستم مبتنی بر حافظه (دیتا به جای ذخیره­ی روی دیسک بر memory یا RAM ذخیره می­گردد) استفاده می­شود. اکثر دیتابیسها اطلاعات را روی دسیک ذخیره می­کنند و به دلیل اینکه مکانیسم خواندن و نوشتن بر روی دیسک­ها فیزیکی است، دسترسی به اطلاعات نیاز به صرف زمان زیادی دارد. به منظور دسترسی سریع به داده­ها از in-memory Databases یا به اختصار IMDB استفاده می­کنیم. (برای دسترسی به آدرسهای مختلف در RAM باید از CPU با قدرت پردازش بالا استفاده کنیم.)

بنابراین هرجا که نیاز به دسترسی سریع، با تعدا فراخوانی بالا به داده­ها وجود داشته باشد، (به عنوان مثال در بانکداری، بازیهای آنلاین تعاملی، پردازش داده­ی سنسورها و ...) از IMDB استفاده می­کنیم. یکی از معایب دیتابیس­های IMDB، پاک شدن یا از دست رفتن دیتا در صورت خاموشی و یا خرابی سرور (به علت ذخیره ی دیتا بر روی RAM) می­باشد. برای رفع این مشکل می توان از مکانیزم­های Transaction logging، Snapshot files و ... استفاده کرد.

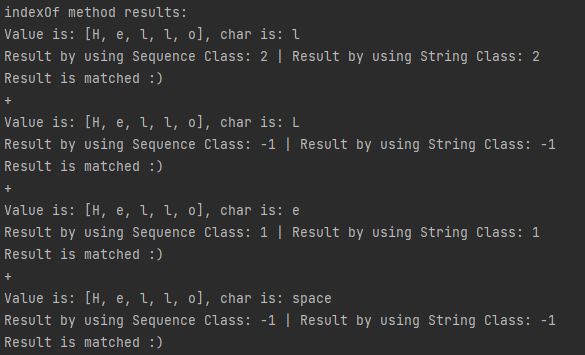
از جمله از دیتابیسهای IMDB می توان به Redis، H2، TaranTool، ArangoDB اشاره کرد.

**بخش دوم**

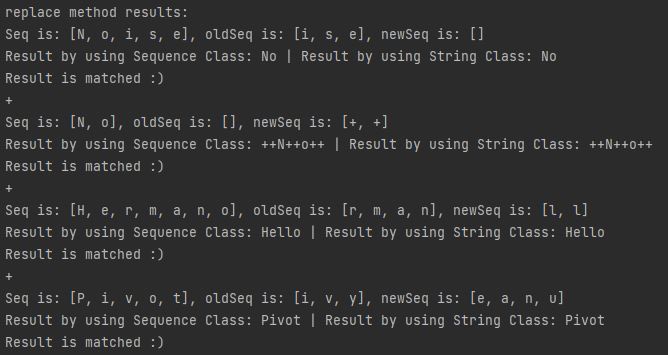
نتیجه­ی تمامی متدهای خواسته شده برای کلاس Sequence با کلاس String تطابق داشته و این قضیه به طور کامل چک شده است (نتایج مربوطه در شکلهای 2، 3، 4 و 5 برای هرکدام از متدها به صورت جداگانه آمده است.)، با این تفاوت که پس از اعمال آنها مقدار المان های آرایه value در کلاس Sequence نیز تغییر می­کند.



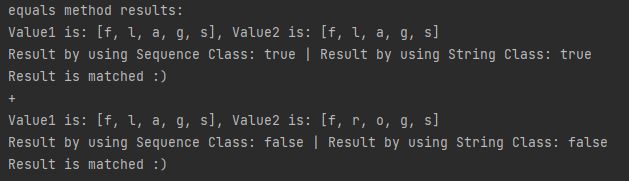
شکل 2 concat method



شکل 3 indexOf method



شکل4 replace method



شکل 5 equals method

9- در این سوال چهار کلاس با نام­های TravelAgency به عنوان آژانس هواپیمایی، FlightTicket به عنوان بلیط پرواز، SalesPerson به عنوان فروشنده و Customer به عنوان مشتری طراحی شده­اند.

10- برای این سوال 8 کلاس تعریف شده است. کلاس­های Stadium برای استادیوم، Team برای هریک از تیم­ها، Ticket برای بلیط بازی، Spectator برای تماشاگرها، Referee برای داور، Player برای هریک از بازیکن­ها، Match برای مسابقه و کلاس Coach برای مربی تعریف شده اند.