Τμήμα Πληροφορικής

Σύγχρονα Θέματα Τεχνολογίας Λογισμικου

Εργασία στο MVC Μοντέλο (2 μονάδες)

Παράδοση: 16 Φεβρουαρίου 2020

Τεχνικό Εγχειρίδιο

Αναπτύχθηκε από τους φοιτητές

Γραμμένο Γεράσιμο Π16025

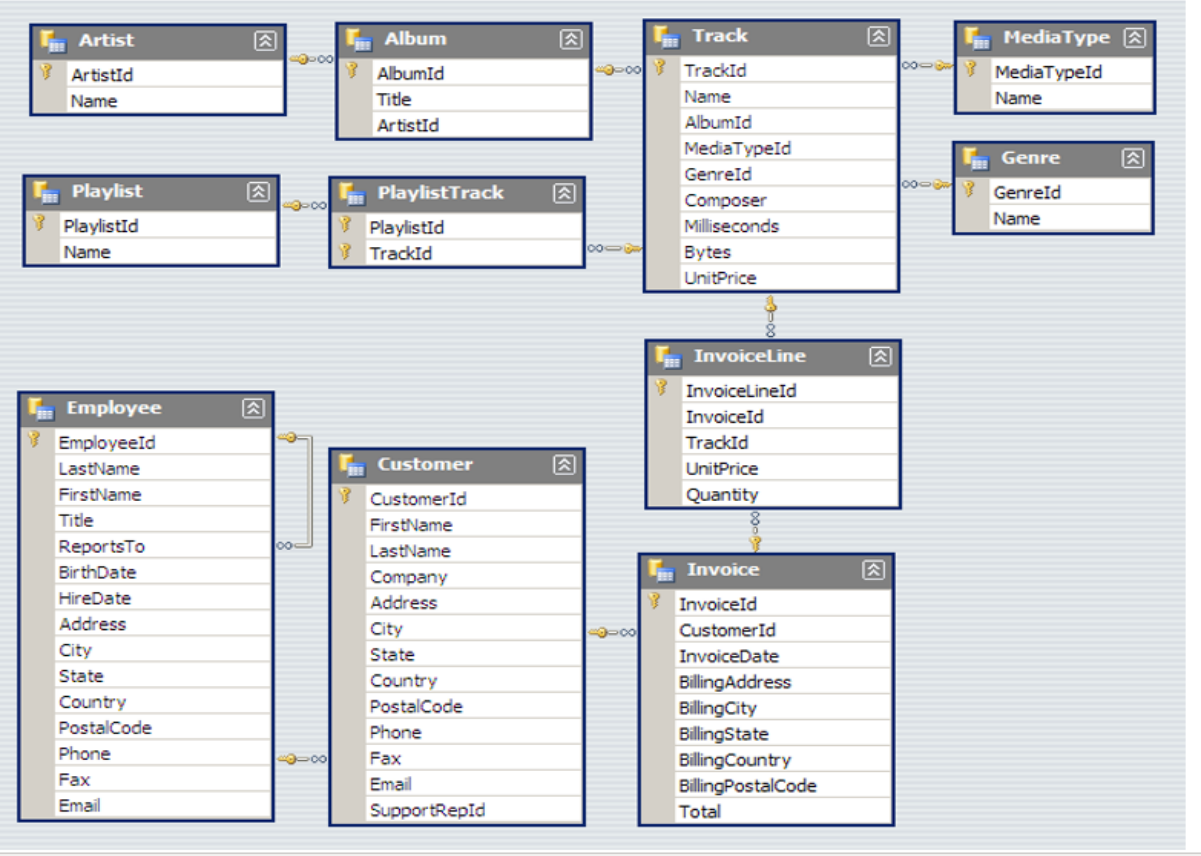
Μηναδάκη Γεώργιο Π13084

Βεργιάννη Νικόλαο Π16170

Σημειώσεις

Όλα τα στιγμιότυπα που παρουσιάζονται στο παρών έγγραφο υπάρχουν στον φάκελο της παραδοτέας εργασίας.

Τα ερωτήματα απαντήθηκαν σε SQL queries στο Microsoft SQL Server Management Studio αφού έγινε restore η Βάση Δεδομένων Chinook από το media file Chinook.bak. Το διάγραμμα της βάσης είναι το παρακάτω:



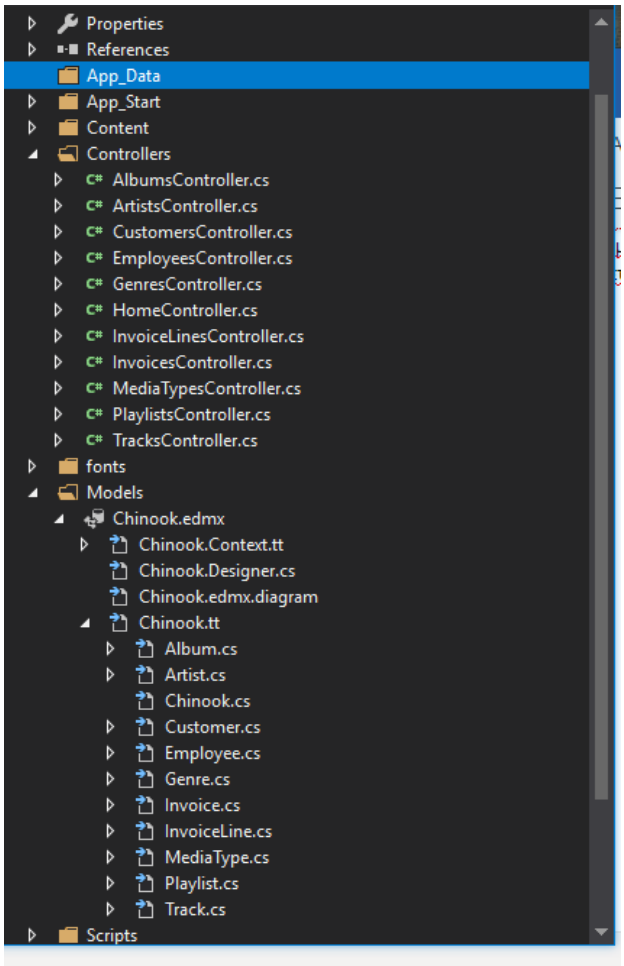
Στην υλοποίηση των views έγινε χρήση **Bootstrap**

<u>Εργαλεία που χρησιμοποιήθηκαν</u>

- Visual Studio 2017 (Asp.Net framework)
- Microsoft SQL Server Managment Studio

Σύνδεση Βάσης Δεδομένων με την εφαρμογη

Εφόσον έχει δημιουργεί η Βάση Δεδομένων στο πρόγραμμα MS SQL Server, κάνουμε εγκατάσταση του Entity Framework και στην σύνεχεια χρήση του ADO.NET Entity Data Model για να δημιουργήσουμε το αντίστοιχο μοντέλο της Βάσης στην εφαρμογή. Δημιουργούμε έναν controller MVC 5 with views για κάθε ένα από τα τραπέζια της βάσης. Όπως φαίνεται στην παρακάτω εικόνα.



Ο κάθε controller**(scaffold)** εμπεριέχει διαδικασίες **CRUD (Create, Read, Update, Delete)** για κάθε καταχώρηση του τραπεζιού της Βάσης Δεδομένων.

Αναζήτηση Καλλιτεχνών που οι δίσκοι τους είναι ανάμεσα στα πρώτα Χ σε πωλήσεις για συγκεκριμένο χρονικό διάστημα

Η αναζήτηση των καλλιτεχνών γίνεται από την σελίδα **Home/Reports/TopArtists**. Πραγματοποιήται ερώτηση στην βάση και στην συνέχεια γυρίζει ο αριθμός των συγγραφέων που ζητήθηκε με την χρήση ενός μετρητή που μετράει τα τραγούδια από κάθε αλμπουμ κάθε καλλιτέχνη, ποιο εμφανίστηκε περισσότερες φορες σε κάθε γραμμη Τιμολογίου(InvoiceLine,Invoice). Με την χρήση του **FirstOrDefault()** διώχνουμε τις duplicate γραμμές.

Η σελίδα κάνει χρήση της τεχνολογίας **Razor C#** και συγκεκριμένα της συνάρτησης **Html.BeginForm** σε συνδυασμό με την μέθοδο **Post**. Εφόσον συμπληρωθεί η φόρμα και πατηθεί το κουμπί **Submit** τα δεδομένα περνάνε με την μέθοδο post σε μία **ActionResult** μέθοδο η οποία έχει το χαρακτηρηστικό **HttpPost** όπως φαίνεται παρακάτω.

```
[HttpPost]
0 references
public ActionResult TopArtists(string StartDate, string EndDate, string ArtistAmount)
{
    DateTime startDate;
    DateTime endDate;
    int artistAmount;



    if (StartDate =="")
    {
        // StartDate in db
        startDate = DateTime.Parse("2009-01-01 00:00:00.000");
    }
    else
    {
        startDate = DateTime.Parse(StartDate);
    }

    if (EndDate == "")
    {
        //end date in db
        endDate = DateTime.Parse("2013-12-22 00:00:00.000");
    }
    else
    {
        endDate = DateTime.Parse(EndDate);
    }

    if (ArtistAmount =="")
    {
        //bigint
        artistAmount = 2144583647;
    }

    else
    {
        artistAmount = Int32.Parse(ArtistAmount);

    }
```

```
    var result = from art in db.Artists
                 join alb in db.Albums on art.ArtistId equals alb.ArtistId
                 join t in db.Tracks on alb.AlbumId equals t.AlbumId
                 join il in db.InvoiceLines on t.TrackId equals il.TrackId
                 join i in db.Invoices on il.InvoiceId equals i.InvoiceId
                 where (i.InvoiceDate >= startDate) && (i.InvoiceDate <= endDate)
                 group art by art.ArtistId into g
                 let count = g.Count()
                 orderby count descending
                 select new
                 {
                     ArtistId = g.Key,
                     Name = g.Select(m => m.Name).FirstOrDefault()

                 };



    var artists = new List<Artist>();

    int counter = 0;
    foreach (var art in result)
    {
        if (counter >= artistAmount) { break; }

       artists.Add(new Artist()
        {
           ArtistId = art.ArtistId,
             Name = art.Name

        });
         counter++;
    }

    return View(artists);

}
```

SQL QUERY:

```
SELECT * FROM Artist
              INNER JOIN
             (SELECT TOP 5 Artist.ArtistId, count(*) as counter
              From Artist
              INNER JOIN Album
              On Artist.ArtistID = Album.ArtistId
              INNER JOIN Track
              ON Album.AlbumId = Track.AlbumId
              INNER JOIN InvoiceLine
              ON Track.TrackId = InvoiceLine.TrackId
              INNER JOIN Invoice
              ON InvoiceLine.InvoiceId = Invoice.InvoiceId
              WHERE InvoiceDate BETWEEN '2009-01-01 00:00:00.000' AND '2013-01-01 00:00:00.000'
              GROUP BY Artist.ArtistId , Artist.Name
              ORDER BY counter DESC) as top5
              ON top5.ArtistId = Artist.ArtistId;
```

100 %

Results | Messages

	ArtistId	Name	ArtistId	counter
1	90	Iron Maiden	90	104
2	150	U2	150	82
3	50	Metallica	50	74
4	22	Led Zeppelin	22	72
5	113	Os Paralamas Do Sucesso	113	36

Αναζήτηση 10 Κορυφαίων τραγουδιών στις προτιμήσεις των πελατών για συγκεκριμένο χρονικό διάστημα.

Η λογική είναι ίδια με την αναζήτηση των καλλιτεχνών. Η αναζήτηση των τραγουδιων γίνεται από την σελίδα **Home/Reports/TopTracks**.

```
[HttpPost]
0 references
public ActionResult TopTracks(string StartDate, string EndDate, string TracksAmount)
{


    DateTime startDate;
    DateTime endDate;
    int tracksAmount;



    TracksAmount = "10";


    if (StartDate == "")
    {
        startDate = DateTime.Parse("2009-01-01 00:00:00.000");
    }
    else
    {
        startDate = DateTime.Parse(StartDate);
    }
```

```
    var result = (from til in db.InvoiceLines
                  join t in db.Tracks on til.TrackId equals t.TrackId
                  join i in db.Invoices on til.InvoiceId equals i.InvoiceId
                  where (i.InvoiceDate >= startDate) && (i.InvoiceDate <= endDate)
                  group t by t.TrackId into g
                  let count = g.Count()
                  orderby count descending
                  select new
                  {
                      TrackId = g.Key,
                            Name = g.Select(m => m.Name).FirstOrDefault(),
                            Album = g.Select(m =>m.Album).FirstOrDefault(),
                            Genre = g.Select(m =>m.Genre).FirstOrDefault(),
                            Composer = g.Select(m =>m.Composer).FirstOrDefault()
                 });


    var tracks = new List<Track>();

    int counter = 0;



    foreach (var tr in result)
    {
        if (counter >= tracksAmount) { break; }

        tracks.Add(new Track()
        {

            TrackId = tr.TrackId,
            Name = tr.Name,
            Album = tr.Album,
            Genre = tr.Genre,
            Composer = tr.Composer

        });

        counter++;
```

SQL QUERY

```
SELECT * FROM TRACK
 INNER JOIN
(SELECT TOP 10  InvoiceLine.TrackId, count(*) as counter
 From InvoiceLine
 INNER JOIN Track
 ON InvoiceLine.TrackId = Track.TrackId
 INNER JOIN Invoice
 ON InvoiceLine.InvoiceId = Invoice.InvoiceId
 WHERE InvoiceDate BETWEEN '2009-01-01 00:00:00.000' AND '2013-12-23 00:00:00.000'
 GROUP BY InvoiceLine.TrackId, Track.Name
 ORDER BY counter DESC) as top10
 ON top10.TrackId = Track.TrackId
 ;
```

100 %

Results | Messages

	TrackId	Name	AlbumId	MediaTypeId	GenreId	Composer	Milliseconds	Bytes	UnitPrice	TrackId	counter
1	925	Last Cup Of Sorrow	74	1	4	Bill Gould/Mike Patton	251663	8221247	0.99	925	2
2	2136	Plot 180	176	1	10	Brian Eno, Bono, Adam Clayton, The Edge & Larry ...	221596	7253729	0.99	2136	2
3	2491	Cherub Rock	202	1	4	Billy Corgan	299389	9786739	0.99	2491	2
4	430	I'm Going Slightly Mad	36	1	1	Queen	248032	8192339	0.99	430	2
5	2322	So Central Rain	190	1	4	R.E.M.	194768	6414550	0.99	2322	2
6	2299	Undertow	189	1	1	Bill Berry-Peter Buck-Mike Mills-Michael Stipe	309498	10131005	0.99	2299	2
7	948	Easy	75	1	4	NULL	185835	6073008	0.99	948	2
8	407	Só Tinha De Ser Com Você	34	1	7	Vários	389642	13085596	0.99	407	2
9	161	Snowblind	17	1	3	Tony Iommi, Bill Ward, Geezer Butler, Ozzy Osbourne	331676	10813386	0.99	161	2
10	1904	The Duke	157	1	2	Dave Brubeck	214961	6977626	0.99	1904	2

Μια άλλη εναλλακτική σε SQL είναι η αναζήτηση των τραγουδιών μέσα από τον πίνακα Playlist.

```
SELECT * FROM TRACK
 INNER JOIN
(SELECT TOP 10  PlaylistTrack.TrackId, count(*) as counter
 From InvoiceLine
 INNER JOIN Track
 ON InvoiceLine.TrackId = Track.TrackId
 INNER JOIN Invoice
 ON InvoiceLine.InvoiceId = Invoice.InvoiceId
 INNER JOIN PlaylistTrack
 ON Track.TrackId = PlayListTrack.TrackId
 INNER JOIN Playlist
 ON PlaylistTrack.PlaylistId = Playlist.PlaylistId
 WHERE InvoiceDate BETWEEN '2009-01-01 00:00:00.000' AND '2013-12-23 00:00:00.000'
 GROUP BY PlaylistTrack.TrackId, Track.Name
 ORDER BY counter DESC) as top10
 ON top10.TrackId = Track.TrackId
```

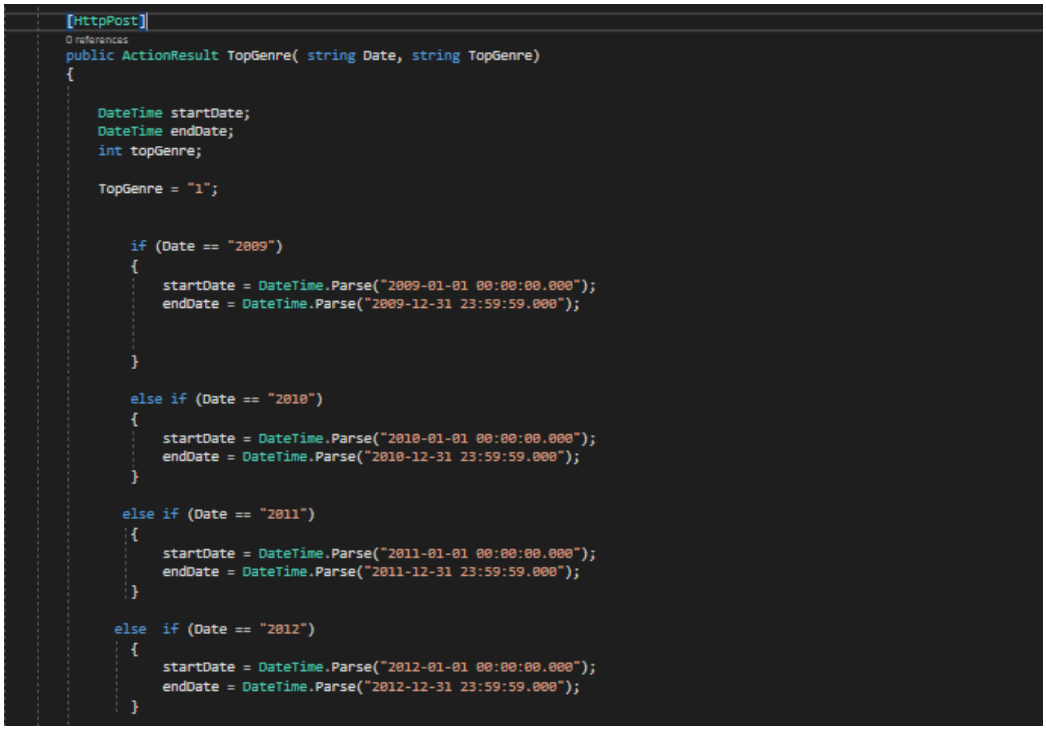
100 %

Results | Messages

	TrackId	Name	AlbumId	MediaTypeId	GenreId	Composer	Milliseconds	Bytes	UnitPrice	TrackId	cour
1	3482	Suite No. 3 in D, BWV 1068: III. Gavotte I & II	327	2	24	Johann Sebastian Bach	225933	3847164	0.99	3482	10
2	3432	Scheherazade, Op. 35: I. The Sea and Sindbad's S...	299	2	24	Nikolai Rimsky-Korsakov	545203	8916313	0.99	3432	10
3	3446	Symphonie Fantastique, Op. 14: V. Songe d'une nu...	312	2	24	Hector Berlioz	561967	9173344	0.99	3446	10
4	858	Esquinas	69	1	7	NULL	280999	9096726	0.99	858	8
5	3500	String Quartet No. 12 in C Minor, D. 703 "Quartetts...	344	2	24	Franz Schubert	139200	2283131	0.99	3500	8
6	3488	Music for the Funeral of Queen Mary: VI. "Thou Kn...	333	2	24	Henry Purcell	142081	2365930	0.99	3488	8
7	867	Açai	70	1	7	Djavan	270968	8893682	0.99	867	8
8	1099	A Novidade (Live)	86	1	7	NULL	316969	10508000	0.99	1099	8
9	925	Last Cup Of Sorrow	74	1	4	Bill Gould/Mike Patton	251663	8221247	0.99	925	6
10	2136	Plot 180	176	1	10	Brian Eno, Bono, Adam Clayton, The Edge & Larry ...	221596	7253729	0.99	2136	6

ΕΙΔΗ ΜΟΥΣΙΚΗΣ ΔΙΑΧΡΟΝΙΚΑ ΠΡΩΤΑ

Η λογική είναι ίδια με την αναζήτηση των καλλιτεχνών. Η αναζήτηση των ειδών μουσικής γίνεται από την σελίδα **Home/Reports/TopGenre.**



```
[HttpPost]
0 references
public ActionResult TopGenre( string Date, string TopGenre)
{

    DateTime startDate;
    DateTime endDate;
    int topGenre;

    TopGenre = "1";


        if (Date == "2009")
        {
            startDate = DateTime.Parse("2009-01-01 00:00:00.000");
            endDate = DateTime.Parse("2009-12-31 23:59:59.000");



        }

        else if (Date == "2010")
        {
            startDate = DateTime.Parse("2010-01-01 00:00:00.000");
            endDate = DateTime.Parse("2010-12-31 23:59:59.000");
        }

       else if (Date == "2011")
        {
            startDate = DateTime.Parse("2011-01-01 00:00:00.000");
            endDate = DateTime.Parse("2011-12-31 23:59:59.000");
        }

      else  if (Date == "2012")
        {
            startDate = DateTime.Parse("2012-01-01 00:00:00.000");
            endDate = DateTime.Parse("2012-12-31 23:59:59.000");
        }
```

```
        else  if (Date == "2012")
            {
                startDate = DateTime.Parse("2012-01-01 00:00:00.000");
                endDate = DateTime.Parse("2012-12-31 23:59:59.000");
            }

        else  if (Date == "2013")
            {
                startDate = DateTime.Parse("2013-01-01 00:00:00.000");
                endDate = DateTime.Parse("2013-12-31 23:59:59.000");
            }

          else
            {
            startDate = DateTime.Parse("");
            endDate = DateTime.Parse("");
            }

             if  (TopGenre == "")
            {
                topGenre = 2144583647;
            }

          else
            {
                topGenre = Int32.Parse(TopGenre);

            }
```

```
    var result = from g in db.Genres
                 join t in db.Tracks on g.GenreId equals t.GenreId
                 join il in db.InvoiceLines on t.TrackId equals il.TrackId
                 join i in db.Invoices on il.InvoiceId equals i.InvoiceId
                 where (i.InvoiceDate >= startDate) && (i.InvoiceDate <= endDate)
                 group g by g.GenreId into g
                 let count = g.Count()
                 orderby count descending
                 select new
                 {
                     GenreId = g.Key,
                     Genre = g.Select(m => m.Name).FirstOrDefault()

                 };


    var genres = new List<Genre>();

    int counter = 0;
    foreach (var g in result)
    {
        if (counter >= topGenre) { break; }

        genres.Add(new Genre()
        {
            GenreId = g.GenreId,
            Name = g.Genre

        });
        counter++;
    }




    return View(genres);
}
```

SQL QUERY

```
SELECT* FROM Genre
                INNER JOIN
                 (SELECT TOP 1 Genre.GenreId, count(*) as counter
                 From Genre
                 INNER JOIN Track
                 ON Genre.GenreId = Track.GenreId
                 INNER JOIN InvoiceLine
                 ON Track.TrackId = InvoiceLine.TrackId
                 INNER JOIN Invoice
                 ON InvoiceLine.InvoiceId = Invoice.InvoiceId
                 WHERE InvoiceDate BETWEEN '2009-01-01 00:00:00.000' AND '2009-12-31 00:00:00.000'
                 GROUP BY Genre.GenreId, Genre.Name
                 ORDER BY counter DESC) as top5
                 ON top5.GenreId = Genre.GenreId;
```

100 %

Results | Messages

	GenreId	Name	GenreId	counter
1	1	Rock	1	180