**Triggers:**

* Αρχικά έχουμε κατασκευάσει έναν trigger (τον sum\_grade\_trig) ο οποίος όπως προαναφέρουμε σε συνδυασμό με την stored procedure “resultSum” που είναι εμφωλευμένη στο σώμα του,ουσιαστικά γεμίζει το πεδίο sum\_grade του πίνακα evaluation\_parts. Πιο αναλυτικά,ο trigger είναι προγραμματισμένος με το που γίνεται εγγραφή στον πίνακα evaluation\_parts, να καλεί την “resultSum” (βλ. Stored Procedures).
* Έπειτα πρέπει μα κατασκευάσουμε triggers οι οποίοι ενημερώνουν τον πίνακα logs, ανάλογα με το αν γίνεται εγγραφή, διαγράφη ή ακόμα και επεξεργασία στους πίνακες Job,Employee και Evaluation\_Request. Για τον κάθε έναν από τους παραπάνω πίνακες θα κατασκευάσουμε 3 triggers εφόσον υπάρχουν 3 δυνατές ενέργειες(insert,delete,update) που μπορούν να υποστούν.
* Για τον πίνακα **Job** λοιπόν, κατασκευάζουμε τους:

1. **Job\_Insert**: Αφού γίνει insert στον πίνακα job , αναγνωρίζει το id της θέσης εργασίας που μόλις προστέθηκε(μέσω της εντολής SELECT INTO) και το τυπώνει-εγγράφει , μαζί με ένα φιλικό προς το χρήστη μήνυμα, στο πεδίο log(TEXT) του πίνακα Logs.
2. **Job\_Delete**: Προτού γίνει delete στον πίνακα job , αναγνωρίζει το id της θέσης εργασίας που επρόκειτο να διαγραφεί (μέσω της εντολής SELECT INTO) και το τυπώνει-εγγραφεί , μαζί με ένα φιλικό προς το χρήστη μήνυμα, στο πεδίο log(TEXT) του πίνακα Logs.
3. **Job\_Update:** Αφού γίνει update στον πίνακα job , ελέγχει αν το id θέσης εργασίας παρέμεινε ίδιο. Αν είναι διαφορετικό τότε αναγνωρίζει το id που μόλις προστέθηκε και το τυπώνει- εγγράφει καθώς και το παλιό id , μαζί με ένα φιλικό προς το χρήστη μήνυμα, στο πεδίο log(TEXT) του πίνακα Logs. Αν το id δεν άλλαξε απλα ενημερώνει τον χρήστη με ένα αντίστοιχο μήνυμα(στον πίνακα Logs) πως υπάρχει κάποια αλλαγή σε κάποιο από τα πεδία της θέσης εργασίας με id=με το id που εχει αναγνωριστεί από τις εντολες SELECT INTO.

* Για τον πίνακα **Employee**, κατασκευάζουμε τους:

1. **Employee\_Insert**: Αφού γίνει insert στον πίνακα employee, αναγνωρίζει το username του υπαλλήλου που μόλις προστέθηκε(μέσω της εντολής SELECT INTO) και το τυπώνει-εγγραφει , μαζί με ένα φιλικό προς το χρήστη μήνυμα, στο πεδίο log(TEXT) του πίνακα Logs.
2. **Employee\_Delete**: Προτού γίνει delete στον πίνακα employee , αναγνωρίζει το username του υπαλλήλου που επρόκειτο να διαγραφεί (μέσω της εντολής SELECT INTO) και το τυπώνει-εγγραφεί , μαζί με ένα φιλικό προς το χρήστη μήνυμα, στο πεδίο log(TEXT) του πίνακα Logs.
3. **Employee\_Update:** Αφού γίνει update στον πίνακα employee, ελέγχει αν το username του υπαλλήλου παρέμεινε ίδιο. Αν είναι διαφορετικό τότε αναγνωρίζει το username που μόλις προστέθηκε και το τυπώνει-εγγραφει καθώς και το παλιό username , μαζί με ένα φιλικό προς το χρήστη μήνυμα, στο πεδίο log(TEXT) του πίνακα Logs. Αν το username δεν άλλαξε απλα ενημερώνει τον χρήστη με ένα αντίστοιχο μήνυμα(στον πίνακα Logs) πως υπάρχει κάποια αλλαγή σε κάποιο από τα πεδία του υπαλλήλου με username=με το username που εχει αναγνωριστεί από τις εντολες SELECT INTO.

* Για τον πίνακα **Evaluation\_Request**, κατασκευάζουμε τους:

1. **Ev\_Request\_Insert**: Αφού γίνει insert στον πίνακα Evaluation\_Request, αναγνωρίζει το username του υπαλλήλου και το id της θέσης εργασίας του αιτήματος που προστέθηκε(μέσω της εντολής SELECT INTO) και τα τυπώνει-εγγραφει , μαζί με ένα φιλικό προς το χρήστη μήνυμα, στο πεδίο log(TEXT) του πίνακα Logs.
2. **Εv\_Request\_Delete:** Προτού γίνει delete στον πίνακα employee , αναγνωρίζει το username του υπαλλήλου και το id της θέσης εργασίας του αιτήματος που επρόκειτο να διαγραφεί (μέσω της εντολής SELECT INTO) και το τυπώνει-εγγραφεί , μαζί με ένα φιλικό προς το χρήστη μήνυμα, στο πεδίο log(TEXT) του πίνακα Logs.
3. **Ev\_Request\_Update:** Αφού γίνει update στον πίνακα employee, ελέγχει αν το username του υπαλλήλου παρέμεινε ίδιο. Αν είναι διαφορετικό τότε αναγνωρίζει το username του εργαζομένου που μόλις εκανε αλλαγή στην αιτηση, και το τυπώνει-εγγραφει, καθώς και το παλιό username , μαζί με ένα φιλικό προς το χρήστη μήνυμα, στο πεδίο log(TEXT) του πίνακα Logs. Αν το username δεν άλλαξε απλα ενημερώνει τον χρήστη με ένα αντίστοιχο μήνυμα(στον πίνακα Logs) πως υπάρχει κάποια αλλαγή σε κάποιο από τα πεδία της αίτησης που έκανε ο εργαζόμενος με username=με το username που εχει αναγνωριστεί από τις εντολες SELECT INTO.

* Στην συνέχεια η εκφώνηση μας ζητά να κατασκευάσουμε έναν trigger , ο οποίος πρακτικά να απαγορεύει την οποιαδήποτε αλλαγή στα πεδια ΑΦΜ,ΔΟΥ και Όνομα του πίνακα **Company**. Για να το πετύχουμε αυτό, θα κατασκευάσουμε τον trigger:
* **Company\_Update:** Στην περίπτωση που γινει update στον πίνακα company, ελεγχει αν οι τιμές των AFM,DOY και Name (που αναγνωρίστηκαν μεσω των εντολών SELECT INTO) έχουν ανανεωθεί. Αν οι τιμές είναι πράγματι διαφορετικές, τότε με την εντολή **SET** τους ανατίθεται η παλιά τους τιμή ενώ τυπώνει στον χρήστη και ένα μήνυμα στο οποίο τον ενημερώνει πως το ΑΦΜ,η ΔΟΥ και το Όνομα θα διατηρήσουν τις τιμές τους.
* Τελος, ζητείται από την εκφώνηση να κατασκευάσουμε ένα trigger το οποίο ουσιαστικά να εξετάζει αν ο χρήστης που κάνει την οποιαδήποτε αλλαγή ή εγγραφή ή διαγραφή,είναι administrator.Στην περίπτωση που δεν είναι administrator ομώς να του απαγορεύει τις παραπάνω ενέργειες.

Εφόσον δεν είχαμε χρόνο όμως να ασχολήθουμε με το δεύτερο κομματι της εργασίας δηλαδη το GUI ο συγκεκριμένος trigger δεν μπορεί να υλοποιηθεί καθώς δεν υπάρχει η οποιαδήποτε “log-in” διαδικασία σε αυτή τη φάση του project.