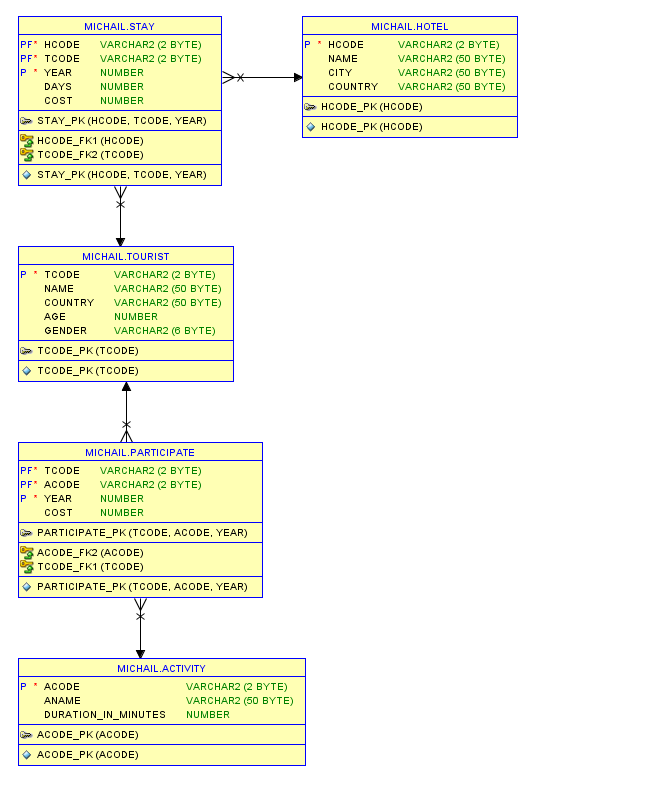
# HOLIDAYS RDBMS

## ERD



### Requirements for Data Model

hotel (hcode, name, city, country);

tourist (tcode, name, country, age, gender);

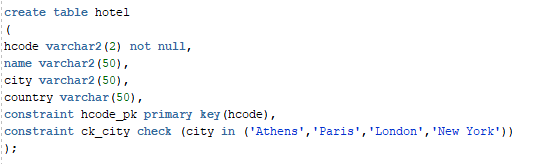
stay (hcode, tcode, year, days, cost);

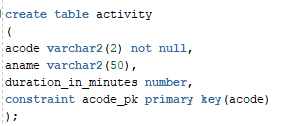
activity (acode, aname, duration\_in\_minutes);

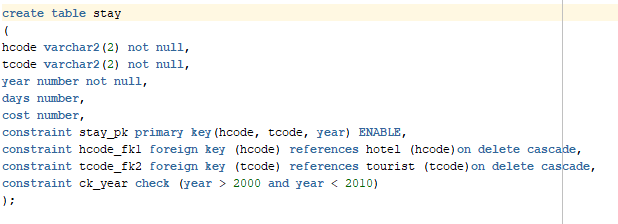
participate (tcode, acode, year, cost);

The Data model above can be created with many different ways. I Choose to use this specific structure while maintaining compound keys.

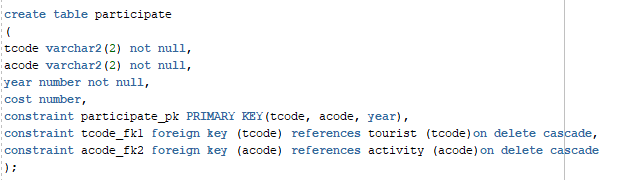
First, we create the tables with necessary constraints checks and foreign keys using **DDL**







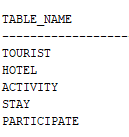
Cascade deletes on child tables the entries/rows if the parent is removed



“Participate” could not had “**year”** column referenced as Foreign key because compound key exists on Parent “Stay” and either you had to include all columns of parent in this table too OR make a new Primary key on this or another table to keep track of.

Then we ensure all tables have been created





We can now safely proceed with **DML**

(portion of inserts below)

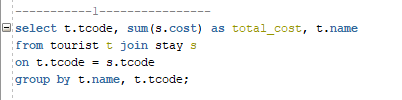


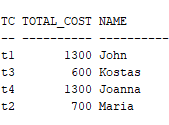
The Database is now created holding model data.

## SQL operations on Data Model

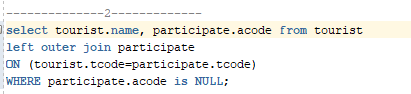
In the HOLIDAYS\_FINAL.sql I included more than 1 way to achieve an output correct result.

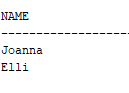
1. Να εμφανιστεί το συνολικο κόστος διαμονης ανά τουρίστα



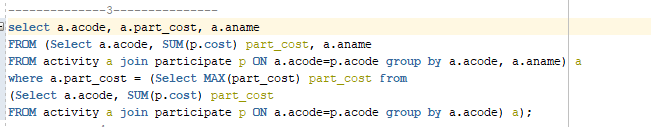


1. Να εμφανιστούν τα ονόματα των τουριστών που δεν έχουν κάνει κάποια δραστηριότητα.



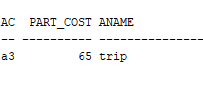


1. Να εμφανιστεί η περιγραφή της δραστηριότητας με τα περισσότερα έσοδα.

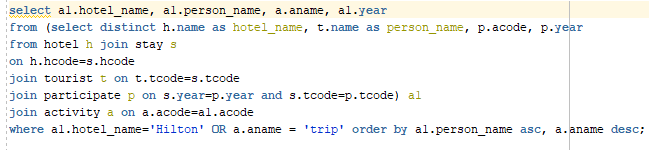


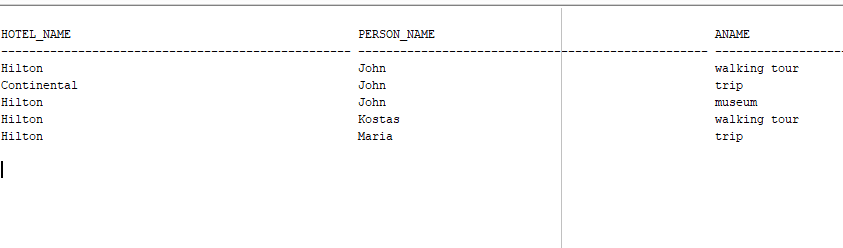
Sub-queries have massive performance hit

Here we create a virtual table called “a” while have created a **Sum** of “costs” then from a.xxx we grabbing fields, again we create another virtual table with the **Max of Sum**



1. Για κάθε τουρίστα που έμεινε στο «Hilton» ξενοδοχείο ή συμμετείχε σε δραστηριότητα «trip» να εμφανιστεί το όνομα του, το όνομα του ξενοδοχείου που έμεινε και το όνομα της δραστηριότητας που συμμετείχε. Τα αποτελέσματα θα πρέπει να εμφανιστούν ταξινομημένα σε αύξουσα σειρά ως προς το όνομα του τουρίστα και σε φθίνουσα σειρά ως προς το όνομα της δραστηριότητας.

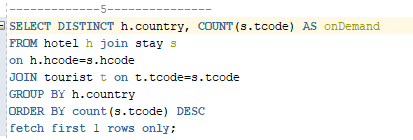


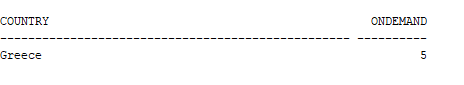


Here we also check “stay year” = “participate year” fields in order to take in account hotels + participate events **OF the SAME YEAR**

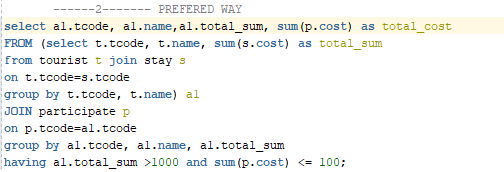
We create a virtual Table “**a1”** (join directly produces extra repeated multiplication of same data with is unnecessary instead this way eliminates this fact) then we join in that virtual table the “activity” which is actually 1 join directly so no extra rows

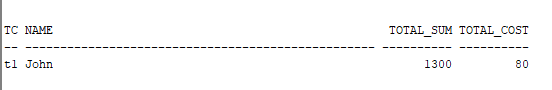
1. Να εμφανιστεί η χώρα την οποία επισκέφτηκαν οι περισσότεροι τουρίστες. Ως χώρα θεωρείστε τη χώρα του ξενοδοχείου που έμειναν. Κάθε τουρίστας να μετρηθεί μόνο μία φορά για κάθε χώρα.





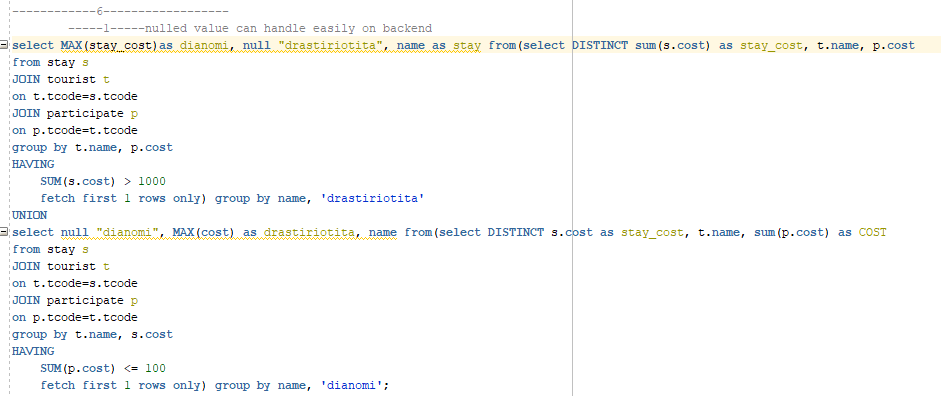
1. Για κάθε τουρίστα που ξόδεψε συνολικά περισσότερα από 1000 ευρώ για διαμονή και 100 ευρώ για δραστηριότητες να εμφανιστεί το όνομα του και το συνολικό ποσό που ξόδεψε για διαμονή και δραστηριότητες.
   1. Way 1 - Preferred

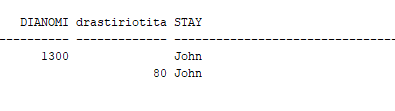




Here compared to question **4** we don’t use “participate year” = “stay year” check because we want total amount. If it had taken in account, it will be no results.

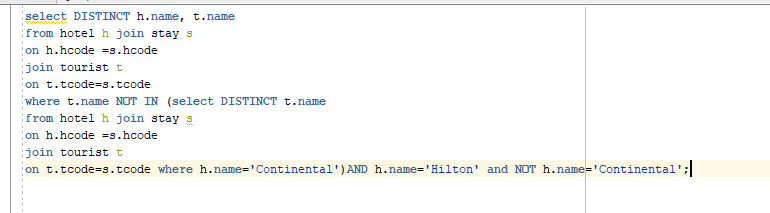
* 1. Way 2 – Hacky way still works Correct

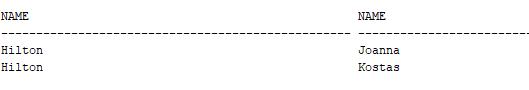




2 Tables Same Columns MUST have so in order to use UNION this approach uses null fields so it is still easy and correct to fetch in a back-end model domain

1. Να εμφανιστούν τα ονόματα των τουριστών που έμειναν στο ξενοδοχείο «Hilton» αλλά όχι στο ξενοδοχείο «Continental».





In the above Result I took in consideration the fact that John and Maria? Had been in “Continental” so we must INCLUDE only the ones that THEY are never been

**The below is wrong**

