**2Ο ΠΡΟΤΖΕΚΤ ΣΧΕΔΙΑΣΜΟΣ ΒΑΣΕΩΝ ΔΕΔΟΜΕΝΩΝ**

ΚΑΡΑΓΕΩΡΓΗΣ ΠΛΑΤΩΝ p3180068

ΖΗΤΗΜΑ 1

Δημιουργώ τον πίνακα campdata όπως φαίνεται παρακάτω:

**Text

Description automatically generated**

Έπειτα, έχοντας κατεβάσει το zip από το eclass φορτώνω τα δεδομένα του .txt στον πίνακα αυτόν με την εντολή που δίνεται στην εκφώνηση.

Στη συνέχεια, το κυριότερο κομμάτι θεωρώ της εργασίας ζητά να σχεδιάσουμε μία αποθήκη δεδομένων που να ακολουθεί το Star Schema. Έχοντας κάνει τα ερωτήματα 1β και 1γ παραθέτω το σχήμα από την επιλογή “Database Diagrams” και στη συνέχεια θα εξηγήσω τη λογική την οποία ακολούθησα προκειμένου να καταλήξω στο αποτέλεσμα της φωτογραφίας.

Diagram

Description automatically generated

Το πρώτο ζητούμενο με τη δημιουργία ενός Star Schema είναι να καταλήξουμε σε ένα fact table. Εγώ ήθελα το fact table να έχει μία σύνοψη όλων των δεδομένων και να έχει κάποιο νόημα, να μην έχει δηλαδή απλά columns που να εξασφαλίζουν την μοναδικότητα των rows με τα δεδομένα που έχουμε ως τώρα. Ο fact table εδώ είναι ο πίνακας “Reservations” δηλαδή ο πίνακας των κρατήσεων. Μία κράτηση, χρησιμοποιώντας την κοινή λογική για να είναι μοναδική και να έχει και τις απαραίτητες πληροφορίες, χρειάζεται τις δύο πλευρές που έρχονται σε συμφωνία, δηλαδή την κατασκήνωση και το τουριστικό γραφείο, τον κωδικό κράτησης, την ημερομηνία που θα πραγματοποιηθεί η διαμονή και το «δωμάτιο». Είναι σχεδιαστικό λάθος να υπάρχουν δύο ίδιες τέτοιες πλειάδες, γιατί αν γινόταν τότε θα ήταν καταστροφικό για τον οικοδεσπότη, στην προκειμένη την κατασκήνωση. Με μία πρώτη ματιά θα χρειαζόμασταν λοιπόν τα custID, bookID, startDate, campCode, empno. Το startDate μας εξασφαλίζει όπως λέει και η εκφώνηση, την ημερομηνία, το empno το «δωμάτιο», το bookID τον κωδικό της κράτησης και τέλος τα custID, campCode είναι οι δύο πλευρές που επέρχονται σε συμφωνία.

Αφού λοιπόν κατέληξα σε αυτό το fact table και τα primary keys του, τότε το μόνο που έμενε ήταν να δημιουργήσω αντίστοιχα dimension tables με primary keys τις τιμές που έχει ορίσει ως foreign το fact table. Το μόνο που είχε μία παραπάνω δυσκολία, ήταν το TIME\_INFO στο οποίο επέλεξα να δώσω αναλυτικά σε κάθε στήλη τα στοιχεία της ημερομηνίας startDate για απλότητα στα μελλοντικά queries. Τέλος, αποφάσισα να προσθέσω τις στήλες overnights, persons, unitCost στον πίνακα Reservations επειδή αφενός μεν θεώρησα ότι ταίριαζαν εκεί περισσότερο αφετέρου δε επιτάχυναν τα queries του ζητήματος 2, καθώς με αυτόν τον τρόπο γλίτωνα περιττά joins.

Παραθέτω τις εντολές που εκτέλεσα στο ζήτημα αυτό:

Text

Description automatically generated with medium confidenceText

Description automatically generatedGraphical user interface, text, application

Description automatically generated

Text

Description automatically generatedText, letter

Description automatically generated

ΖΗΤΗΜΑ 2

1)

SELECT TOP 100 country, fname, lname, SUM(unitCost\*persons\*overnights) AS total\_cost

FROM TOURIST\_OFFICE, RESERVATIONS

WHERE RESERVATIONS.custID = TOURIST\_OFFICE.custID

GROUP BY country, fname, lname

ORDER BY total\_cost DESC;

2)

SELECT campName, category, SUM(unitCost\*persons\*overnights) AS total\_cost

FROM RESERVATIONS, CAMP, ROOMS, TIME\_INFO

WHERE RESERVATIONS.empno = ROOMS.empno AND RESERVATIONS.campCode = CAMP.campCode AND RESERVATIONS.startDate = TIME\_INFO.startDate AND t\_year = 2000

GROUP BY campName, category

ORDER BY campName;

3)

SELECT campName,t\_month, SUM(unitCost\*persons\*overnights) AS total\_cost

FROM RESERVATIONS, CAMP, ROOMS, TIME\_INFO

WHERE RESERVATIONS.empno = ROOMS.empno AND RESERVATIONS.campCode = CAMP.campCode AND RESERVATIONS.startDate = TIME\_INFO.startDate AND t\_year = 2018

GROUP BY campName, t\_month

ORDER BY campName;

4)

SELECT t\_year, campName, category, SUM(persons) as total\_tenants

FROM RESERVATIONS,TIME\_INFO, CAMP, ROOMS

WHERE RESERVATIONS.startDate = TIME\_INFO.startDate AND RESERVATIONS.campCode = CAMP.campCode AND RESERVATIONS.empno = ROOMS.empno

GROUP BY ROLLUP(t\_year,campName,category);

5)

CREATE VIEW [PER\_YEAR\_TENANTS] AS

SELECT campName, t\_year, SUM(persons) as total\_tenants

FROM RESERVATIONS,TIME\_INFO, CAMP

WHERE RESERVATIONS.startDate = TIME\_INFO.startDate AND RESERVATIONS.campCode = CAMP.campCode

GROUP BY campName, t\_year;

SELECT DISTINCT table1.campName

FROM [PER\_YEAR\_TENANTS] as table1, [PER\_YEAR\_TENANTS] as table2

WHERE table1.campName = table2.campName AND table1.t\_year = 2018 AND table2.t\_year = 2017 AND (table1.total\_tenants > table2.total\_tenants);

ΖΗΤΗΜΑ 3

Το query που θα χρησιμοποιήσω για το ζήτημα 3 είναι το παρακάτω:

SELECT RESERVATIONS.campCode, catCode, t\_year, SUM(persons) as total\_tenants

FROM RESERVATIONS, CAMP, ROOMS, TIME\_INFO

WHERE RESERVATIONS.campCode = CAMP.campCode AND RESERVATIONS.empno = ROOMS.empno AND RESERVATIONS.startDate = TIME\_INFO.startDate

GROUP BY CUBE(RESERVATIONS.campCode,catCode,t\_year);

Α) Παραθέτω ένα screenshot από τις διαλέξεις για να εξηγήσω την απάντηση μου:

Diagram

Description automatically generated

Αν υποθέσουμε ότι o ένας άξονας αντί για Product έχει Rooms, ο δεύτερος άξονας αντί για Quarter έχει Camp και ο τρίτος άξονας αντί για Region έχει τον πίνακα Time\_Info, τότε το κελί της φωτογραφίας θα περιέχει π.χ. το σύνολο των tenants στο δωμάτιο με catCode A, στη κατασκήνωση με campCode APL και το έτος 2000. Δηλαδή, κάθε κελί είναι η τιμή του column που βρίσκεται στο aggregate function, η οποία προκύπτει από τον συνδυασμό των τριών τιμών στις διαστάσεις x, y, x του κύβου.

Β) Τα GROUP BY είναι συνολικά 8 διότι ο κύβος έχει 3 διαστάσεις και άρα 2^3=8 περιπτώσεις. Αυτές τις βλέπουμε παρακάτω:

1Η Περίπτωση : campCode, catCode ,t\_year (GROUP BY campCode ,catCode, t\_year)

2Η Περίπτωση : NULL, catCode, t\_year (GROUP BY catCode, t\_year)

3Η Περίπτωση : campCode, NULL ,t\_year (GROUP BY campCode , t\_year)

4Η Περίπτωση : campCode, catCode, NULL (GROUP BY campCode ,catCode)

5Η Περίπτωση : NULL, NULL, t\_year (GROUP BY t\_year)

6Η Περίπτωση : campCode, NULL, NULL (GROUP BY campCode)

7Η Περίπτωση : NULL, catCode , NULL (GROUP BY catCode)

8Η Περίπτωση : NULL, NULL, NULL (GROUP BY NONE)

ΖΗΤΗΜΑ 4

Το bitmap για όλες τις χώρες της στήλης countries θα έχει την ακόλουθη μορφή:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Finland | Sweden | Italy | Holland | Germany | Austria | France | Belgium | Spain | Denmark |
| Row |  |  |  |  |  |  |  |  |  |  |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 10 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

Με βάση αυτή την «κωδικοποίηση», το περιεχόμενο του ευρετηρίου για τις 8 δειγματοληπτικές εγγραφές που δίνει η εκφώνηση θα είναι:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Finland | Sweden | Italy | Holland | Germany | Austria | France | Belgium | Spain | Denmark |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 4 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |