Isys2120 week 12 lab

A(i). Indexing

- Declaring the table UnitOfStudy(uoSCode, deptId, uoSName, credits) will give a primary key (indegrated, clustered) on uoSCode.
- This means we can efficiently find the row when we are given a particular uoSCode value
  - Get to the data of the row without scanning all the other rows
  - Just follow pointers in the index, to the data of that row
  - So this is efficient for a query such as

  SELECT * FROM UnitOfStudy WHERE uoSCode = ‘ISYS2120’;

A(ii)

- But what about a query

```
SELECT uoSName
FROM UnitOfStudy
WHERE deptId = 'SIT';
```

The primary index is no help here, because we don't know the uoSCode of the units we are seeking!

So the only approach (with default physical structure) is to scan all the rows, checking each one to see whether it has the deptId we want

- This is very expensive if the table is big

A(iii)

- Declare an index on deptId

CREATE INDEX uos_deptId_ix ON UnitOfStudy(deptId);

Now, we can find the data for rows with any particular deptID, using the index, without scanning the whole table

An alternative would be an index on a composite of several columns, as long as deptId is the first of the columns in the index. To cover the query, we can execute instead

CREATE INDEX uos_deptId_ix1 ON UnitOfStudy(deptId, uoSName);

Now both deptId and uoSName are in the index, so we don't need to check the data row at all

- However, the index will take more space, than the single-column index

B(i)

```
SELECT M.region, P.category,
SUM(S.Sales_amt)
FROM Sales S, Market M, Product P
WHERE S.Market_Id = M.Market_Id
       and S.Product_Id =
P.Product_Id
GROUP BY CUBE(M.region, P.category)
```

- Result (relational format, from PostgreSQL)

REGION	CATEGORY	SUM(S.SALES_AMT)
		109372
	Meat	6078
	Drink	45374
	Soft Goods	57920
East		10471
East	Meat	5673
East	Drink	2850
East	Soft Goods	1948
West		98901
West	Meat	405
West	Drink	42524
West	Soft Goods	55972

Marginal, showing region='West',
aggregating over all categories as well as all times

B(i)

```
SELECT M.region, P.category,
SUM(S.Sales_amt)
FROM Sales S, Market M, Product P
WHERE S.Market_Id = M.Market_Id
       and S.Product_Id =
P.Product_Id
GROUP BY CUBE(M.region, P.category)
```

- Result (cube format)

SUM(S.SALES_AMT)	East	West	Total
Meat	5673	405	6078
Drink	2850	42524	45374
Soft Goods	1948	55972	57920
Total	10471	98901	109372

Marginal, showing region='West',
aggregating over all categories as well as all times

B(ii)

- Slice for a given week, dicing by Product_Id and State

```
SELECT S.Product_Id, M.state, SUM(S.Sales_amt)
FROM Sales S, Time T, Market M
WHERE S.Market_id = M.Market_Id
        and S.Time_Id = T.Time_Id
        and T.week = 'Wk-05-2006'
Group by cube(S.Product_Id, M.state)
```