

# CSE 111 – DATABASE SYSTEMS

## Final Exam

Consider the following relational schema:

**Product** (model, type, maker)

**Distributor** (model, name, price)

**Price\_Cube** (distributor\_type, product\_type, num\_prod, tot\_price)

in which the keys are underlined. *Distributor.model* is a foreign key to *Product.model*. Referential integrity is enforced through CASCADE operations.

**Product** and **Distributor** are base tables, while **Price\_Cube** is a data cube with two dimensions – *distributor\_type* and *product\_type* – and two measure attributes—*num\_prod* and *tot\_price*. *distributor\_type* takes only two values – “producer” and “distributor” – where “producer” corresponds to a product maker, while “distributor” to an entity that does not make any product. *product\_type* corresponds to attribute *type* from **Product**, which can be only one of {“pc”, “laptop”, “printer”}. *num\_prod* is the total number of products of a given type sold by “producer”/“distributor”, while *tot\_price* is the total price corresponding to these products. Given these, we know that **Price\_Cube** has exactly 12 tuples, including the value “ALL” or “\*”.

Sample data for the base tables **Product** and **Distributor** is given below and is also included in the database `data.sqlite`, which is provided to you. The data cube **Price\_Cube** generated over these sample data is also included below (you have to build it).

			<i>model</i>	<i>name</i>	<i>price</i>				
			100001	P1	NULL				
			100002	A	2874				
			100003	A	NULL				
			100004	P3	NULL				
<i>model</i>	<i>type</i>	<i>maker</i>	100005	P1	878	<i>dist_type</i>	<i>prod_type</i>	<i>num_prod</i>	<i>tot_price</i>
100001	pc	P1	200001	P4	534	ALL	ALL	21	16934
100002	pc	A	200002	B	998	ALL	laptop	6	6828
100003	pc	A	200003	P1	NULL	ALL	pc	10	9429
100004	pc	P3	200004	P3	NULL	ALL	printer	5	677
100005	pc	P1	300001	P6	NULL	distributor	ALL	9	11363
200001	laptop	P4	300002	P6	NULL	distributor	laptop	2	5296
200002	laptop	B	300003	A	287	distributor	pc	5	5677
200003	laptop	P1	100001	D1	1239	distributor	printer	2	390
200004	laptop	P3	100003	D2	1645	producer	ALL	12	5571
300001	printer	P6	100004	D2	674	producer	laptop	4	1532
300002	printer	P6	100003	Z	1245	producer	pc	5	3752
300003	printer	A	100004	Z	874	producer	printer	3	287
			200003	D1	1875				
			200004	Y	3421				
			300001	D2	223				
			300002	D2	167				

It is important to notice that the **maker** of a product is always a **distributor** of the product, even when the **price** is unknown, i.e., NULL. This constraint has to be maintained at all times as follows. For every tuple in **Product**, there is a corresponding tuple in **Distributor** having the same value for **model** and **name** in **Distributor** is equal to **maker** in **Product**. The value of **price** is set to NULL by default. INSERT on

**Distributor** with the **maker** of a product is treated as an UPDATE of the **price** attribute. DELETE from **Distributor** with the **maker** of a product is treated as an UPDATE of the **price** to NULL.

Given the skeleton code in `Final.java` and `Final.py`, you have to implement the following methods/functions:

- `print_Product` prints the tuples in the **Product** table.
- `print_Distributor` prints the tuples in the **Distributor** table.
- `build_data_cube` builds the **Price\_Cube** data cube from the base tables.
- `print_Cube` prints the tuples in **Price\_Cube**.
- `modifications` performs the INSERT and DELETE operations specified in file `modifications.txt`. There is an operation specified on every line. The operation gives the table on which the operation is performed, followed by the operation type (I or D), and the arguments of the operation. When executing these operations, all the constraints have to be satisfied. While we provide you sample operations, your code will be tested on a different set of operations. Thus, it has to be general. However, the only operations are INSERT and DELETE on **Product** and **Distributor**, respectively. Remember to keep the data cube **Price\_Cube** consistent with the data in the base tables **Product** and **Distributor** as these are modified.

Your code has to be written in the above methods/functions. Additionally, you can create whichever methods/functions you find necessary. However, you cannot change the `main` method/function. Your code will be tested on an instance of the database `data.sqlite` by running the file `./test.sh` in the terminal (this file is provided). We will check the output of your code based on different input databases `data.sqlite` and modification files `modifications.txt`. Moreover, make sure that your code is safe to SQL injection attacks. Finally, your submission will be run through a plagiarism detection software.

The only file you have to submit is `Final.java` or `Final.py`, whichever you decide to implement. Do not include any other files in your submission because we will not consider them. We will exclusively grade your code with `./test.sh` in the terminal.