### **//Second draft**

### **// select 1:**

### // find the type of plugin that downloaded the most

### mysql> select type from plugins **join** (

### select plugin, count from (

### select plugin, count(plugin) as count from downloaded group by plugin) as d1

### where d1.count >= all (

### select **count**(plugin) from downloaded **group by** plugin))d2

### where d2.plugin = plugins.plugin;

### answer:

### +------+

### | type |

### +------+

### FX

### +------+

**// select 2**

// find user who downloaded both fx and generator plugins

select name from user where uid = (select uID from (select uID, plugin from downloaded natural join plugins where plugin != all (select name from fx)) t1 **join**

(select uID, plugin from downloaded natural join plugins where plugin = any (select name from fx)) t2 **using**(uID));

+------+

| name |

+------+

| John |

+------+

**//select 3**

// find developers who make both fx and generator plugins and list their total plugins in the plugin table.

select developer, **count**(developer) as numOfPlugins from plugins old

where type !=any (select type from plugins where developer = **old.developer**) **group by** (developer);

+------------+--------------+

| developer | numOfPlugins |

+------------+--------------+

| Kilohearts | 37 |

| XFER | 4 |

| TAL | 3 |

+------------+--------------+

### 

### **// trigger 1:**

// create trigger check user must be an existing user to download plugin and update download //times in user table attributes.

delimiter //

drop trigger if exists validateUserForDownload;

create trigger validateUserForDownload

before insert on downloaded

for each row

if new.uID in (select uID from user)

then update User set downLoaded = downLoaded +1 where uID = new.uID;

end if;//

delimiter ;

// trigger 2:

// delete record from downloaded table, the user table downloaded plugin’s time will be reduced // as well, this trigger will not check delete record input for validation. So it is allow delete //multiple record, as long as the value in the where clause exist in the downloaded table

delimiter //

drop trigger if exists deleteDownloadRecord;

create trigger deleteDownloadRecord

before delete on downloaded

for each row

begin

update User set downloaded = downloaded -1 where uid = old.uid;

end;//

delimiter ;

**// delete 1**: work with trigger 2

delete from downloaded where uid = 7 and plugin ='Haas' and downloadDate='2021-09-26';

mysql> select \* from user;

+-----+-------+------+------------+

| uID | name | age | downLoaded |

+-----+-------+------+------------+

| 1 | Kim | 20 | 2 |

| 2 | John | 24 | 2 |

| 3 | Amy | 20 | 1 |

| 4 | Lucas | 28 | 3 |

| 5 | Lio | 19 | 1 |

| 6 | Mike | 30 | 3 |

| 7 | Mia | 21 | 1 | the reorder in user table update from 2 to 1 after delete from download table

+-----+-------+------+------------+

**// delete 2**: work with trigger 2

delete from downloaded where uid = 7;

mysql> select \* from user;

+-----+-------+------+------------+

| uID | name | age | downLoaded |

+-----+-------+------+------------+

| 1 | Kim | 20 | 2 |

| 2 | John | 24 | 2 |

| 3 | Amy | 20 | 1 |

| 4 | Lucas | 28 | 3 |

| 5 | Lio | 19 | 1 |

| 6 | Mike | 30 | 3 |

| 7 | Mia | 21 | 0 | after delete all record of uid 1, now this user show downloaded times: 0

+-----+-------+------+------------+

**// insert 1: work with the above trigger**

mysql> insert into downloaded values(1, 'Haas', now());

Query OK, 1 row affected (0.01 sec)

mysql> select \* from user;

+-----+-------+------+------------+

| uID | name | age | downLoaded |

+-----+-------+------+------------+

| 1 | Kim | 20 | 2 | // now kim has 2 downloaded

| 2 | John | 24 | 2 |

| 3 | Amy | 20 | 1 |

| 4 | Lucas | 28 | 3 |

| 5 | Lio | 19 | 1 |

| 6 | Mike | 30 | 3 |

| 7 | Mia | 21 | 2 |

+-----+-------+------+------------+

**// insert 2**

### **// stored procedure:**

// stored procedure

// remove old plugins from plugins, fx and generator table and store in archive

drop procedure if exists archive;

delimiter //

create procedure archive(IN last\_update timestamp)

begin

insert into archive (name, format, developer, type, subtype, isFree, hasLicense, lastUpdate, uID, downloadDate) select name, format, developer, type, subtype, isFree, hasLicense, lastUpdate, uID, downloadDate from plugins join fx on fx.name = plugins.plugin and fx.lastUpdate < last\_update left outer join downloaded on downloaded.plugin = plugins.plugin;

insert into archive (name, format, developer, type, subtype, isFree, hasLicense, lastUpdate, uID, downloadDate) select name, format, developer, type, subtype, isFree, hasLicense, lastUpdate, uID, downloadDate from plugins join generator on generator.name = plugins.plugin and generator.lastUpdate < last\_update left outer join downloaded on downloaded.plugin = plugins.plugin;

delete from plugins where plugins.plugin in (select name from fx where lastUpdate < last\_update);

delete from plugins where plugins.plugin in (select name from generator where lastUpdate < last\_update);

delete from fx where lastUpdate < last\_update;

delete from generator where lastUpdate < last\_update;

end //

delimiter ;

call archive('2020-01-01');

#I will add a few more select features

#1.

#select the names of the plugins that have been downloaded

#from Fx and Generator after the year 2020

#select distinct generator.name from generator

#left outer join downloaded on generator.name = downloaded.plugin

#where generator.lastUpdate > '2019-12-31' and downloaded.downloadDate > '2019-12-31'

#union all

#select distinct fx.name from fx

#right outer join downloaded on fx.name = downloaded.plugin

#where fx.lastUpdate > '2019-12-31' and downloaded.downloadDate > '2019-12-31';

#2.

#select number of generator plugins and fx plugins in VST3 format

#

#select count(distinct generator.name) as genCount, count(distinct fx.name) as fxCount

#from generator, FX

#where generator.format = 'VST3' and fx.format = 'VST3';

#3. select average number of plugins in fx table that are free

#select avg(stcounts.ct)

#from(

#select count(subtype) as ct

#from fx

#where fx.isFree = 1

#group by subtype

#having count(subtype) > 1

#) as stcounts;

#4. delete plugins that have not been updated in 4 months

#trying to find way to combine these two statements

#feel free to edit

delete from generator

where generator.lastUpdate > generator.lastUpdate + '0000-04-00';

delete from fx

where fx.lastUpdate > generator.lastUpdate + '0000-04-00';

### **// first draft**

### **Report Requirements**

1. Team name: CPTW
2. Members name:
3. Project title: PLUGINS
4. Complete database schema with constraints, also showing the relation(s) to be archived.
5. 15 functional requirements and associated SQL. there are at least 5 significantly different queries involving different relations and attributes.
   * **Please highlight in bold the five siginificantly different types of SQL (one co-related subquery, group by and having, aggregation, outer join, and mathematical set operation)**
   * List all SQL select statements

// select developers who develop at least two generator plugins that are both free and have a license.

select name from developer d where dID in (select dID from generator where isFree = 1 and hasLicense = 1 and d.dID = dID **group by dID having count(dID)**>1 )

// select count generator with subtype of Softsynth where lastUpdate is after 2021

select count(\*) from generator where subtype = "Softsynth" and lastUpdate > 2020-12-31;

* + List all SQL update statements
  + List all SQL delete statements

// delete generators with lastUpdate earlier than 2021-01-01

delete from Generator where lastUpdate < 2021-01-01;

* + List all SQL insert statements
  + List all SQL triggers

// move generator plugins to oldpluins which lastupdate are before 2021

DROP Trigger moveToOld;

delimiter //

CREATE Trigger moveToOld

after delete on Generator

for each row

begin

insert into OldPlugins (old.pID, old.dID, old.name, old.format, old.type, old.subtype, old.isFree, old.hasLicense, old.lastUpdate);

end//

delimiter;

* + List all SQL stored procedures
  + Screenshots to show that at least three different user requests can be taken to the running **Java application** and then be served successfully.