# Trig.sql document

## First Trigger: MODIFY_HERO_LEVEL

This trigger will calculate a hero's level based on the amount of experience they have accrued. It will react to the creation, update, or deletion of a value from the 'exp' column of the Hero table. A change will be applied to the 'HLEVEL' table, reflecting a change in the hero's level. The level is calculated as $\log_2$ experience.

Here is an example that could fire the trigger:

```
INSERT INTO hero VALUES (101, 'Batman', TO_DATE('02-12-2016','DD-MM-YYYY'), 8888);
UPDATE hero SET exp = 7777 where pid = 101;
DELETE FROM hero WHERE pid = 101;
```

## Second Trigger: MODIFY_WEAPONVALUE

This trigger will calculate a weapon's value based on its physical damage, magical damage, and rarity. The trigger is run when there is a creation, update, or deletion applied to the Weapon table's 'mdamage', 'pdamage', or 'rarity' attributes. The weights of the physical and magical damage on the weapon's value are dependent on its rarity attribute.

Here is an example that could fire the trigger:

```
INSERT INTO weapon VALUES (101, ' Sword1', 2, 0, 'Common', 1, 'BarMoneyPack', 'Save the Princess');
UPDATE weapon SET mdamage = 111 WHERE wid = 101;
DELETE FROM weapon WHERE wid = 101;
```

## Third Trigger: MODIFY_MONSTERHP

This trigger will calculate a monster's hit points based on its level. The trigger runs when there is an addition, update, or deletion applied to the 'MHP' table, reflecting a change in the monster's hitpoints. A monster's hitpoints are calculated as 1.2 * level + 100.

Here is an example that could fire the trigger:

INSERT INTO monster VALUES(‘Hello kitty’, 100,’Castle of Doom’ );
UPDATE monster SET mlevel = 200 WHERE mname=’Hello Kitty’;
DELETE FROM monster WHERE mname = ‘Hello Kitty’;

## Fourth Trigger: CHECK_HERO_EXP

This trigger will run before a insertion or update is applied to the ‘exp’ column of the ‘HERO’ table. If this value is negative, an exception is thrown.

Here is an example that could fire the trigger:
INSERT INTO hero VALUES (101, 'Batman', TO_DATE('02-12-2016','DD-MM-YYYY'), 8888);
UPDATE hero set exp = -1 where pid = 101;

## Fifth Trigger: CHECK_MON_LEVEL

This trigger will run before a insertion or update is applied to the ‘mlevel’ column of the ‘MONSTER’ table. If this value is negative, an exception is thrown.

Here is an example that could fire the trigger:

INSERT INTO monster VALUES('Hello kitty',100,'Castle of Doom');
UPDATE monster SET mlevel = -1 WHERE mname='Hello Kitty';

Appendix:

```
CREATE OR REPLACE TRIGGER CHECK_HERO_EXP
BEFORE INSERT OR UPDATE OF exp ON hero
FOR EACH ROW
DECLARE
        neg_exp EXCEPTION;
BEGIN
        IF :NEW.exp < 0 THEN
                RAISE neg_exp;
        END IF;
EXCEPTION
        WHEN neg_exp THEN
        RAISE_APPLICATION_ERROR(-20001,'Cannot set hero experience negative numnber');
```

```
END;
/

CREATE OR REPLACE TRIGGER CHECK_MON_LEVEL
BEFORE INSERT OR UPDATE OF mlevel ON monster
FOR EACH ROW
DECLARE
        neg_level EXCEPTION;
BEGIN
        IF :NEW.mlevel < 0 THEN
                RAISE neg_level;
        END IF;
EXCEPTION
        WHEN neg_level THEN
        RAISE_APPLICATION_ERROR(-20001,'Cannot set monster level negative numnber');
END;
/


CREATE OR REPLACE TRIGGER modify_herolevel
AFTER INSERT OR UPDATE OR DELETE OF exp ON hero
FOR EACH ROW
BEGIN
   IF INSERTING THEN
        INSERT INTO HLEVEL VALUES(:NEW.exp,  CAST (log(2,:NEW.exp) AS INT));
   ELSIF UPDATING THEN
     DELETE FROM HLEVEL WHERE hexp = :OLD.exp;
     INSERT INTO HLEVEL VALUES(:NEW.exp, CAST (log(2,:NEW.exp) AS INT));
   ELSIF deleting then
     DELETE FROM HLEVEL WHERE hexp = :OLD.exp;
   END IF;
```

```
END;
/


CREATE OR REPLACE TRIGGER modify_monsterhp
AFTER INSERT OR UPDATE OR DELETE OF mlevel ON monster
FOR EACH ROW
BEGIN
   IF INSERTING THEN
     INSERT INTO MHP VALUES(:NEW.mlevel,  CAST( (100+1.2*:NEW.mlevel) AS INT));
   ELSIF UPDATING THEN
     DELETE FROM MHP WHERE mlevel = :OLD.mlevel;
     INSERT INTO MHP VALUES(:NEW.mlevel, CAST ( (100+1.2*:NEW.mlevel) AS INT));
   ELSIF deleting then
     DELETE FROM MHP   WHERE mlevel = :OLD.mlevel;
   END IF;
END;
/



CREATE OR REPLACE TRIGGER modify_weaponvalue
AFTER INSERT OR UPDATE OR DELETE OF  rarity, pdamage, mdamage ON weapon
FOR EACH ROW
BEGIN
   IF INSERTING THEN
     if :NEW.rarity='Unique' then
     INSERT INTO WVALUE
VALUES(:NEW.rarity,:NEW.mdamage,:NEW.pdamage,CAST( (:NEW.mdamage *
1.1+:NEW.pdamage*1.2+100) AS INT));
     elsif :NEW.rarity = 'Common'then
     INSERT INTO WVALUE
VALUES(:NEW.rarity,:NEW.mdamage,:NEW.pdamage,CAST( (:NEW.mdamage *
1.1+:NEW.pdamage*1.2+10) AS INT));
     elsif :NEW.rarity = 'Rare' then
```

```
    INSERT INTO WVALUE
VALUES(:NEW.rarity,:NEW.mdamage,:NEW.pdamage,CAST( (:NEW.mdamage *
1.1+:NEW.pdamage*1.2+50) AS INT));

    elsif :NEW.rarity = 'Magic' then

    INSERT INTO WVALUE
VALUES(:NEW.rarity,:NEW.mdamage,:NEW.pdamage,CAST( (:NEW.mdamage *
1.1+:NEW.pdamage*1.2+20) AS INT));

    end if;

  ELSIF UPDATING THEN

    DELETE FROM WVALUE WHERE  rarity=:OLD.rarity and damageM=:OLD.mdamage and
damageP=:OLD.pdamage;

    if :NEW.rarity='Unique' then

    INSERT INTO WVALUE
VALUES(:NEW.rarity,:NEW.mdamage,:NEW.pdamage,CAST( (:NEW.mdamage *
1.1+:NEW.pdamage*1.2+100) AS INT));

    elsif :NEW.rarity = 'Common'then

    INSERT INTO WVALUE
VALUES(:NEW.rarity,:NEW.mdamage,:NEW.pdamage,CAST( (:NEW.mdamage *
1.1+:NEW.pdamage*1.2+10) AS INT));

    elsif :NEW.rarity = 'Rare' then

    INSERT INTO WVALUE
VALUES(:NEW.rarity,:NEW.mdamage,:NEW.pdamage,CAST( (:NEW.mdamage *
1.1+:NEW.pdamage*1.2+50) AS INT));

    elsif :NEW.rarity = 'Magic' then

    INSERT INTO WVALUE
VALUES(:NEW.rarity,:NEW.mdamage,:NEW.pdamage,CAST( (:NEW.mdamage *
1.1+:NEW.pdamage*1.2+20) AS INT));

    end if;

  ELSIF deleting then

    DELETE FROM WVALUE WHERE  rarity=:OLD.rarity and damageM=:OLD.mdamage and
damageP=:OLD.pdamage;

  END IF;

END;

/
```