A6: Indexes, triggers, user functions and population

This artefact contains the database's workload as well as the complete database creation script, including all SQL necessary to define all integrity constraints, indexes and triggers.

1. Database Workload

1.1. Tuple Estimation

Relation reference	Relation Name	Order of magnitude	Estimated growth
R01	User	thousands	dozens per day
R02	Customer	thousands	dozens per day
R03	Moderator	units	units per year
R04	Administrator	units	units per year
R05	Banned	dozens	units per month
R06	Commentary	hundreds	units per day
R07	Answer	dozens	units per day
R08	Flagged	dozens	units per

			month
R09	Product	hundreds	units per month
R10	Attribute	dozens	no growth
R11	AttributeProduct	thousands	dozens per month
R12	CatAtt	dozens	no growth
R13	Category	units	no growth
R14	Favorite	thousands	dozens per day
R15	Purchase	thousands	dozens per day
R16	PurchaseProduct	thousands	dozens per day
R17	Rating	hundreds	units per month

1.2. Frequent Queries

Query reference	SELECT01
Query description	Discounted Products
Query frequency	Thousands per day

```
          SELECT product.title,
                 category."name",
                 product.price,
                 product.discountPrice,
            FROM product, category
          WHERE discountPrice != NULL AND
product.idCat = category.id;
```

Query reference	SELECT02
Query description	Products from category
Query frequency	Thousands per day

```
        SELECT product.title,
                category."name",
                product.price,
                product.discountPrice,
          FROM product, category
          WHERE product.idCat = $cat AND
category.id = $cat;
```

Query reference	SELECT03
Query description	Search products by title and comment
Query frequency	Thousands per day

```
      SELECT product.title,
        category."name",
        product.price,
        product.discountPrice,
      FROM product, category
      WHERE search @@
```

```
plainto_tsquery('english',$name) and category.id =
product.idCat;
```

Query reference	SELECT04
Query description	View my profile
Query frequency	Hundreds per day

```
      SELECT
"name","address",loyaltyPoints,email,username,picture
        FROM "user" JOIN customer ON username;
```

Query reference	SELECT05
Query description	View my wish list products
Query frequency	Hundreds per day

```
        SELECT product.title,
                category."name",
                product.price,
                product.discountPrice,
          FROM "user" JOIN favorite ON username,
product, category
          WHERE favorite.refProduct = product.sku;
```

Query reference	SELECT06
Query description	View a product
Query frequency	Thousands per day

```
        SELECT product.sku,
                product.title,
                category."name",
                product.price,
                product.discountPrice,
                product.rating,
                attribute."name",
                attribute_product."value",
          FROM product, attribute_product,
category_attribute, attribute, category
          WHERE product.idCat = category.id
           AND category_attribute.idCategory =
product.idCat
           AND category_attribute.idAttribute =
attribute_product.idAttribute
           AND attribute_product.refProduct =
product.sku;
```

Query reference	SELECT07
Query description	View comments by date
Query frequency	Hundreds per day

```
        SELECT *
        FROM comment
        WHERE "date" < $threshold
```

Query reference	SELECT08
Query description	View the customer's purchase history
Query frequency	Hundreds per day

```
        SELECT purchase."date",
                purchase."value",
                purchase.method,
                purchase_product.idProduct,
                purchase_product.price,
                purchase_product.quantity,
        FROM purchase, purchase_product
        WHERE username = $user,
purchase_product.idPurchase = purchase.id;
```

Query reference	SELECT09
Query description	Search in comments
Query frequency	Hundreds per day

```
         SELECT *
         FROM comment
         WHERE search @@
plainto_tsquery('english',$name);
```

1.3. Frequent Modifications

Query reference	INSERT01
Query description	Write or answer a comment
Query frequency	Dozens per day

```
        INSERT INTO comment(username,commentary)
          VALUES ($username,$commentary);
        INSERT INTO answer(idParent)
          VALUES ($idParent);
```

Query reference	INSERT02
Query description	Mark a product as favourite
Query frequency	Hundreds per day

```
          INSERT INTO favorites(username,refProduct)
            VALUES ($username,$refproduct);
```

Query reference	INSERT03
Query description	Purchase Products
Query frequency	Dozens per day

```
        INSERT INTO purchase

VALUES(DEFAULT,$username,now(),$cost,$method)
        INSERT INTO purchase_product(idPurchase
,idProduct, price, quantity)

VALUES($idPurchase,$idProduct,$price,$quantity);
```

Query reference	INSERT04
Query description	Sign in a new User
Query frequency	Units per day

```
        INSERT INTO "user"
          VALUES($username, $password, $email,
DEFAULT, $picture);
        INSERT INTO customer
          VALUES($username, $name, $address,
$loyaltyPoints, $newsletter, $inactive);
        --OR
        INSERT INTO moderator
          VALUES($username);
        --OR
        INSERT INTO administrator
          VALUES($username);
```

Query reference	INSERT05

Query description	Rate a product
Query frequency	Dozens per day

```
INSERT INTO rating
  VALUES($username,$refProduct,$value);
```

Query reference	UPDATE01
Query description	Update my profile
Query frequency	Dozens per day

```
UPDATE "user"
  SET "password" = $password,
        email = $email,
        picture = $picture
  WHERE username = $username;
UPDATE customer
  SET name = $name,
        address = $address,
        loyaltyPoints = $loyaltyPoints,
        newsletter = $newsletter,
        inactive = $inactive
  WHERE username = $username;
```

Query reference	UPDATE02
Query description	Rate a Product

Query frequency	Dozens per day

```
          UPDATE rating
            SET "value" = $value
            WHERE username = $username AND refProduct
  = $refProduct;
```

2. Proposed Indices

2.1. Performance Indices

Index reference	IDX01
Related queries	SELECT01, SELECT02
Index relation	product
Index attribute	idCat
Index type	Hash
Cardinality	Low
Clustering	No
Justification	Query SELECT01 has to be fast as it is executed many times by people wanting to know the discounted products. Query SELECT02 has to be fast as well to present products by their category. Low cardinality because there aren't many different

	types of categories. We predict no relevant interference on updates or inserts to the purchase product.

```
    CREATE INDEX discounted_product ON product
USING hash(idCat);
```

Index reference	IDX02
Related queries	SELECT07
Index relation	comment
Index attribute	date
Index type	Btree
Cardinality	High
Clustering	No
Justification	Query SELECT07 helps present the most recent comments by a given threshold every time a customer views a product and wants to check other opinions. We predict no relevant interference on updates or inserts to the comment table.

```
    CREATE INDEX comments_range ON comment USING
```

```
btree("date");
```

Index reference	IDX03
Related queries	SELECT08
Index relation	purchase
Index attribute	username
Index type	Hash
Cardinality	Medium
Clustering	No
Justification	Query SELECT08 has to be fast because the user has to quickly check what has bought before. Medium cardinality because of the balance between uniqueness of username among users and many purchases being made by people. We predict no relevant interference on updates or inserts to the purchase table.

```
    CREATE INDEX customer_purchases ON purchase
USING hash(username);
```

2.2. Full-text Search Indexes

Index reference	IDX04
Related queries	SELECT03
Index relation	product
Index attribute	search
Index type	GIN
Clustering	No
Justification	To improve the performance of text searches while searching for products by their title/description. The size and time it takes to generate these type of indexes as opposed to GiST can be seen as a tradeoff because of it's lossless nature. We are predicting few alterations on the products after the database is populated.

```
    CREATE INDEX search_product ON product USING
GIN (search);
```

Index reference	IDX05
Related queries	SELECT09

Index relation	comment
Index attribute	search
Index type	GiST
Clustering	No
Justification	To improve the performance of text searches while searching for comments. Because we are predicting more updates and inserts on the comment table we chose GiST over GIN because of their time tradeoff, being GiST faster and less memory intensive, with the downside of some acceptable loss.

```
    CREATE INDEX search_comments ON comment USING
GIST (search);
```

3. Triggers

Trigger reference	TRIGGER01
Trigger description	User can only be banned after joining.

```
    CREATE OR REPLACE FUNCTION check_banned_date()
RETURNS trigger AS $check_banned_date$
    BEGIN
    IF EXISTS (
        SELECT U.joinDate FROM "user" U  WHERE
U.username = NEW.username_customer AND U.joinDate >
NEW.bannedDate
    )
    THEN RAISE EXCEPTION '% cannot be banned before
joining', NEW.username_customer;
    END IF;

    RETURN NEW;
END;
$check_banned_date$ LANGUAGE plpgsql;

CREATE  TRIGGER check_banned_date BEFORE INSERT OR
UPDATE ON banned
  FOR EACH ROW EXECUTE PROCEDURE check_banned_date();
```

Trigger reference	TRIGGER02
Trigger description	Date of comment answer must be later than parent comment.

```
      CREATE OR REPLACE FUNCTION
check_answer_date() RETURNS trigger AS
$check_answer_date$
      BEGIN
```

```
    IF EXISTS (
        SELECT C1.id, C2.id FROM comment C1, comment
C2
        WHERE
            C1.id < C2.id
            AND
            C1.id = NEW.idParent
            AND
            C2.id = NEW.idChild
            AND
            C1."date" > C2."date"
    )
    THEN RAISE EXCEPTION 'Must comment on an older
commentary.';
    END IF;

    RETURN NEW;
  END;
  $check_answer_date$ LANGUAGE plpgsql;

  CREATE TRIGGER check_answer_date BEFORE INSERT OR
UPDATE ON answer
    FOR EACH ROW EXECUTE PROCEDURE
check_answer_date();
```

Trigger reference	TRIGGER03
Trigger description	Update the rating of a product.

```
```

```
CREATE OR REPLACE FUNCTION update_product_rating()
```

```
RETURNS trigger AS $update_product_rating$
BEGIN

    UPDATE product SET rating = (
        SELECT AVG("value") FROM rating R WHERE
R.refProduct = NEW.refProduct
    )
    WHERE id = NEW.refProduct;

    RETURN NEW;
END;
$update_product_rating$ LANGUAGE plpgsql;

CREATE TRIGGER update_product_rating AFTER INSERT OR
UPDATE ON rating
  FOR EACH ROW EXECUTE PROCEDURE
update_product_rating();
```

Trigger reference	TRIGGER04
Trigger description	Discount price is always lower than price.

```
    CREATE OR REPLACE FUNCTION
constraint_product_discount() RETURNS trigger AS
$constraint_product_discount$
    BEGIN
        IF NEW.discountPrice > NEW.price THEN
        RAISE EXCEPTION 'Discount price must be
lower than price.';
        END IF;
    RETURN NEW;
END;
$constraint_product_discount$ LANGUAGE plpgsql;

CREATE TRIGGER constraint_product_discount BEFORE
INSERT OR UPDATE ON product
  FOR EACH ROW EXECUTE PROCEDURE
constraint_product_discount();
```

Trigger reference	TRIGGER05
Trigger description	Reflect Full Text Search functionality in product.

```
    CREATE OR REPLACE FUNCTION
insert_update_product() RETURNS trigger AS
$insert_update_product$
    BEGIN
```

```
IF TG_OP = 'INSERT'
THEN NEW.search =
setweight(to_tsvector(coalesce(NEW.title,'')), 'A')
||

setweight(to_tsvector(coalesce(NEW.description,'')),
'B');
END IF;
IF TG_OP = 'UPDATE' THEN
    IF NEW.title <> OLD.title OR NEW.decription <>
OLD.description
```

```
    THEN NEW.search =
setweight(to_tsvector(coalesce(NEW.title,'')), 'A')
||

setweight(to_tsvector(coalesce(NEW.description,'')),
'B');
    END IF;
END IF;
RETURN NEW;
END;
$insert_update_product$ LANGUAGE plpgsql;


CREATE TRIGGER insert_update_product BEFORE INSERT OR
UPDATE ON product
  FOR EACH ROW EXECUTE PROCEDURE
insert_update_product();
```

Trigger reference	TRIGGER06
Trigger description	Reflect Full Text Search functionality in comment.

```
    CREATE OR REPLACE FUNCTION
insert_update_comment() RETURNS trigger AS
$insert_update_comment$
    BEGIN
```

```
IF TG_OP = 'INSERT'
THEN NEW.search = to_tsvector(NEW.commentary);
END IF;
IF TG_OP = 'UPDATE' THEN
```

```
    IF NEW.commentary <> OLD.commentary
    THEN NEW.search = to_tsvector(NEW.commentary);
    END IF;
END IF;
RETURN NEW;
END;
$insert_update_comment$ LANGUAGE plpgsql;


CREATE TRIGGER insert_update_comment BEFORE INSERT OR
UPDATE ON comment
  FOR EACH ROW EXECUTE PROCEDURE
insert_update_comment();
```

4. Complete SQL Code

```
--TABLES
CREATE TABLE "user" (
    username text PRIMARY KEY,
    "password" text NOT NULL,
    email text UNIQUE NOT NULL,
    joinDate TIMESTAMP DEFAULT now() NOT NULL,
    picture text
);

CREATE TABLE customer (
    username text PRIMARY KEY REFERENCES "user" ON
DELETE CASCADE,
    "name" text NOT NULL,
    "address" text,
    loyaltyPoints INTEGER NOT NULL DEFAULT 0,
    newsletter BOOLEAN NOT NULL DEFAULT TRUE,
    inactive BOOLEAN NOT NULL DEFAULT FALSE,

    CONSTRAINT lp_positive CHECK ((loyaltyPoints >=
0))
```

```
);

CREATE TABLE moderator (
    username text PRIMARY KEY REFERENCES "user" ON
DELETE CASCADE
);

CREATE TABLE administrator (
    username text PRIMARY KEY REFERENCES "user" ON
DELETE CASCADE
);

CREATE TABLE banned (
    username_customer TEXT PRIMARY KEY REFERENCES
customer ON DELETE CASCADE,
    bannedDate TIMESTAMP DEFAULT now() NOT NULL,
    username_moderator TEXT NOT NULL REFERENCES
moderator ON DELETE CASCADE
);

CREATE TABLE comment (
    id SERIAL PRIMARY KEY,
    username TEXT NOT NULL REFERENCES "user" ON
DELETE CASCADE,
    "date" TIMESTAMP DEFAULT now() NOT NULL,
    commentary text NOT NULL,
    flagsNo INTEGER NOT NULL DEFAULT 0,
    deleted BOOLEAN DEFAULT FALSE NOT NULL,
    refProduct INTEGER NOT NULL REFERENCES product ON
DELETE CASCADE
);

CREATE TABLE answer (
    idParent INTEGER NOT NULL REFERENCES comment ON
DELETE CASCADE,
    idChild INTEGER NOT NULL REFERENCES comment ON
DELETE CASCADE,

    UNIQUE(idParent, idChild)
```

```
);

CREATE TABLE flagged (
    idComment INTEGER NOT NULL REFERENCES comment ON
DELETE CASCADE,
    "hidden" BOOLEAN NOT NULL
);

CREATE TABLE category (
    id SERIAL PRIMARY KEY,
    "name" text NOT NULL
);

CREATE TABLE product (
    sku SERIAL PRIMARY KEY,
    title text NOT NULL,
    idCat INTEGER NOT NULL REFERENCES category ON
DELETE CASCADE,
    price REAL NOT NULL,
    discountPrice REAL,
    rating REAL NOT NULL,
    stock INTEGER NOT NULL,

    CONSTRAINT price_positive CHECK (price > 0),
    CONSTRAINT discount_positive CHECK (discountPrice
is NULL or discountPrice > 0),
    CONSTRAINT stock_positive CHECK(stock >= 0),
    CONSTRAINT rating_positive CHECK(rating >= 0)
);

CREATE TABLE attribute (
    id SERIAL PRIMARY KEY,
    "name" text NOT NULL
);

CREATE TABLE attribute_product (
    idAttribute INTEGER NOT NULL REFERENCES attribute
ON DELETE CASCADE,
    refProduct INTEGER NOT NULL REFERENCES product ON
```

```
DELETE CASCADE,
    "value" text NOT NULL
);

CREATE TABLE category_attribute (
    idAttribute INTEGER NOT NULL REFERENCES attribute
ON DELETE CASCADE,
    idCategory INTEGER NOT NULL REFERENCES category
ON DELETE CASCADE,

    UNIQUE(idAttribute, idCategory)
);

CREATE TABLE favorite (
    username INTEGER NOT NULL REFERENCES customer ON
DELETE CASCADE,
    refProduct INTEGER NOT NULL REFERENCES product ON
DELETE CASCADE,

    UNIQUE(username, refProduct)
);

CREATE TABLE purchase (
    id SERIAL PRIMARY KEY,
    username TEXT NOT NULL REFERENCES customer ON
DELETE CASCADE,
    "date" TIMESTAMP DEFAULT now() NOT NULL,
    "value" REAL NOT NULL,
    method text NOT NULL,

    CONSTRAINT value_positive CHECK ("value" > 0),
    CONSTRAINT method_check CHECK (method in
('Credit', 'Debit' , 'Paypal' ))
);

CREATE TABLE purchase_product (
    idPurchase INTEGER NOT NULL REFERENCES purchase
ON DELETE CASCADE,
    idProduct INTEGER NOT NULL REFERENCES product ON
```

```
DELETE CASCADE,
    price REAL NOT NULL,
    quantity INTEGER NOT NULL,

    CONSTRAINT quantity_positive CHECK (quantity >
0),
    CONSTRAINT price_positive CHECK (price > 0),

    UNIQUE(idPurchase, idProduct)
);

CREATE TABLE rating (
    username text PRIMARY KEY REFERENCES customer ON
DELETE CASCADE
    refProduct INTEGER NOT NULL REFERENCES product ON
DELETE CASCADE,
    "value" INTEGER NOT NULL CHECK (("value" > 0 )
AND ("value" <= 5)),
    PRIMARY KEY(username, refProduct)
);

--QUERIES
SELECT product.title,
        product.idCat,
        product.category,
        product.price,
        product.discountPrice,
  FROM product, category
  WHERE discountPrice != NULL AND product.idCat =
category.id;

SELECT product.title,
        category."name",
        product.price,
        product.discountPrice,
  FROM product, category
  WHERE Product.idCat = $cat AND category.id = $cat;

SELECT product.title,
        category."name",
        product.price,
        product.discountPrice,
  FROM product, category
  WHERE product.title LIKE %$name% and category.id =
product.idCat;

SELECT product.title,
        category."name",
        product.price,
        product.discountPrice,
  FROM "user" JOIN favorite ON username, product,
category
  WHERE favorite.refProduct = product.sku AND
category.id = product.idCat;

SELECT product.sku,
        product.title,
        category."name",
        product.price,
        product.discountPrice,
        product.rating,
        attribute."name",
        attribute_product."value",
  FROM product, attribute_product,
category_attribute, attribute, category
  WHERE product.idCat = category.id
   AND category_attribute.idCategory = product.idCat
   AND category_attribute.idAttribute =
attribute_product.idAttribute
   AND attribute_product.refProduct = product.sku;

SELECT
"name","address",loyaltyPoints,email,username,picture
  FROM "user" JOIN customer ON username;

SELECT *
  FROM comment
  WHERE "date" < $threshold
```

```
SELECT purchase."date",
        purchase."value",
        purchase.method,
        purchase_product.idProduct,
        purchase_product.price,
        purchase_product.quantity,
  FROM purchase, purchase_product
  WHERE username = $user, purchase_product.idPurchase
= purchase.id;

--UPDATES
INSERT INTO comment(username,commentary)
  VALUES ($username,$commentary);
INSERT INTO answer(idParent)
  VALUES (idParent);

INSERT INTO favorites(username,refProduct)
  VALUES ($username,$refproduct);

INSERT INTO purchase
  VALUES(DEFAULT,$username,now(),$cost,$method)
INSERT INTO purchase_product(idPurchase ,idProduct,
price, quantity)
  VALUES($idPurchase,$idProduct,$price,$quantity);

INSERT INTO rating
  VALUES($username,$refProduct,$value);

INSERT INTO "user"
  VALUES($username, $password, $email, DEFAULT,
$picture);
INSERT INTO customer
  VALUES($username, $name, $address, $loyaltyPoints,
$newsletter, $inactive);
INSERT INTO moderator
  VALUES($username);
INSERT INTO administrator
  VALUES($username);
```

```
UPDATE "user"
  SET "password" = $password,
        email = $email,
        picture = $picture
  WHERE username = $username;
UPDATE customer
  SET name = $name,
        address = $address,
        loyaltyPoints = $loyaltyPoints,
        newsletter = $newsletter,
        inactive = $inactive
  WHERE username = $username;

UPDATE rating
  SET "value" = $value
  WHERE username = $username AND refProduct =
$refProduct;

--TRIGGERS
-- 1 user can only be banned after joining

CREATE OR REPLACE FUNCTION check_banned_date()
RETURNS trigger AS $check_banned_date$
    BEGIN

        IF EXISTS (
            SELECT U.joinDate FROM "user" U  WHERE
U.username = NEW.username_customer AND U.joinDate >
NEW.bannedDate
        )
        THEN RAISE EXCEPTION '% cannot be banned
before joining', NEW.username_customer;
        END IF;

        RETURN NEW;
    END;
$check_banned_date$ LANGUAGE plpgsql;
```

```
CREATE  TRIGGER check_banned_date BEFORE INSERT OR
UPDATE ON banned
    FOR EACH ROW EXECUTE PROCEDURE
check_banned_date();


--2 date of comment answer must be later than parent
comment

CREATE OR REPLACE FUNCTION check_answer_date()
RETURNS trigger AS $check_answer_date$
    BEGIN

        IF EXISTS (
            SELECT C1.id, C2.id FROM comment C1,
comment C2
            WHERE
                C1.id < C2.id
                AND
                C1.id = NEW.idParent
                AND
                C2.id = NEW.idChild
                AND
                C1."date" > C2."date"
        )
        THEN RAISE EXCEPTION 'Must comment on an
older commentary.';
        END IF;

        RETURN NEW;
    END;
$check_answer_date$ LANGUAGE plpgsql;

CREATE TRIGGER check_answer_date BEFORE INSERT OR
UPDATE ON answer
    FOR EACH ROW EXECUTE PROCEDURE
check_answer_date();
```

```
-- 3 update the rating of a product

CREATE OR REPLACE FUNCTION update_product_rating()
RETURNS trigger AS $update_product_rating$
    BEGIN

        UPDATE product SET rating = (
            SELECT AVG("value") FROM rating R WHERE
R.refProduct = NEW.refProduct
        )
        WHERE id = NEW.refProduct;

        RETURN NEW;
    END;
$update_product_rating$ LANGUAGE plpgsql;


CREATE TRIGGER update_product_rating AFTER INSERT OR
UPDATE ON rating
    FOR EACH ROW EXECUTE PROCEDURE
update_product_rating();

-- 4 discount price on a product

CREATE OR REPLACE FUNCTION
constraint_product_discount() RETURNS trigger AS
$constraint_product_discount$
    BEGIN
        IF NEW.discountPrice > NEW.price THEN
        RAISE EXCEPTION 'Discount price must be lower
than price.';
        END IF;

        RETURN NEW;
    END;
$constraint_product_discount$ LANGUAGE plpgsql;

CREATE TRIGGER constraint_product_discount BEFORE
INSERT OR UPDATE ON product
```

```
    FOR EACH ROW EXECUTE PROCEDURE
constraint_product_discount();


-- 5 update insert product

CREATE OR REPLACE FUNCTION insert_update_product()
RETURNS trigger AS $insert_update_product$
    BEGIN

    IF TG_OP = 'INSERT'
    THEN NEW.search =
setweight(to_tsvector(coalesce(NEW.title,'')), 'A')
||

setweight(to_tsvector(coalesce(NEW.description,'')),
'B');
    END IF;
    IF TG_OP = 'UPDATE' THEN
        IF NEW.title <> OLD.title OR NEW.decription
<> OLD.description
        THEN NEW.search =
setweight(to_tsvector(coalesce(NEW.title,'')), 'A')
||

setweight(to_tsvector(coalesce(NEW.description,'')),
'B');
        END IF;
    END IF;
    RETURN NEW;
    END;
$insert_update_product$ LANGUAGE plpgsql;


CREATE TRIGGER insert_update_product BEFORE INSERT OR
UPDATE ON product
    FOR EACH ROW EXECUTE PROCEDURE
insert_update_product();
```

```
-- 6 update insert comment

CREATE OR REPLACE FUNCTION insert_update_comment()
RETURNS trigger AS $insert_update_comment$
    BEGIN

    IF TG_OP = 'INSERT'
    THEN NEW.search = to_tsvector(NEW.commentary);
    END IF;
    IF TG_OP = 'UPDATE' THEN
        IF NEW.commentary <> OLD.commentary
        THEN NEW.search =
to_tsvector(NEW.commentary);
        END IF;
    END IF;
    RETURN NEW;
    END;
$insert_update_comment$ LANGUAGE plpgsql;


CREATE TRIGGER insert_update_comment BEFORE INSERT OR
UPDATE ON comment
    FOR EACH ROW EXECUTE PROCEDURE
insert_update_comment();

--INDEXES
DROP INDEX IF EXISTS discounted_product;
CREATE INDEX discounted_product ON product USING
hash(idCat);

DROP INDEX IF EXISTS search_product;
CREATE INDEX search_product ON product USING GIN
(search);

DROP INDEX IF EXISTS search_comments;
CREATE INDEX search_comments ON product USING GIST
(search);
```

```
DROP INDEX IF EXISTS comments_range;
CREATE INDEX comments_range ON comment USING
btree("date");

DROP INDEX IF EXISTS customer_purchases;
CREATE INDEX customer_purchases ON purchase USING
hash(username);

--POPULATE

--USER
INSERT INTO "user" VALUES ('TYTTbQf','1[W-
^c1y!*D','Sharon@dignissim.net','2018-04-03
18:47:05');
INSERT INTO "user" VALUES
('KXiAxRukz','*!WC{bpyrn)h','Gillian@neque.us','2018-
04-10 17:27:54');
INSERT INTO "user" VALUES
('5zkyn6n5U','A)R','Cheyenne@dolor.org','2018-04-14
02:14:13');
INSERT INTO "user" VALUES ('CgsHHRf','V5m:_?
NE_<','Barclay@pellentesque.us','2018-04-05
00:57:24');
INSERT INTO "user" VALUES
('j5zBk','^D+;>C','Fuller@fringilla.edu','2018-04-18
22:41:26');
INSERT INTO "user" VALUES
('0MM8g8','q9!K>J]vH!5%Q#','Britanney@ante.com','2018
-04-17 05:56:50');
INSERT INTO "user" VALUES
('Ssuwu','>`ft','Cheyenne@sapien.net','2018-04-06
20:10:09');
INSERT INTO "user" VALUES
('PjujYy','q]!n%e_[GDV','Patricia@accumsan.us','2018-
04-02 10:02:27');
INSERT INTO "user" VALUES
('glbGcBfG','}3jR.1Dvgy','Naomi@Vestibulum.us','2018-
03-29 19:34:49');
INSERT INTO "user" VALUES
```

```
('l1mtgBKN','T)>W*uUm.]6g','Daria@eros.com','2018-04-
01 12:17:27');
INSERT INTO "user" VALUES
('NRb3xxj','^F1Gg<lUA*','Austin@Nulla.com','2018-04-
16 12:17:10');
INSERT INTO "user" VALUES
('s53d41qTPk',']_hJuw:4IH0nx4k','Jolene@bibendum.edu',
'2018-03-28 10:21:29');
INSERT INTO "user" VALUES
('soMeMGrGpb','vj}r##;:','Montana@rutrum.gov','2018-
04-17 17:48:52');
INSERT INTO "user" VALUES ('UaUyY7HgDC','NSi:79-
','Sebastian@faucibus.gov','2018-04-11 06:25:30');
INSERT INTO "user" VALUES
('x2CW','9<8t:jIdNFJ','Shay@Aliquam.us','2018-04-10
08:36:29');
INSERT INTO "user" VALUES
('104zBsdD','lLzq','Colt@sem.org','2018-03-31
15:47:03');
INSERT INTO "user" VALUES
('n1BRuWgdR','y6AY','Kylynn@tellus.com','2018-03-30
04:58:49');
INSERT INTO "user" VALUES
('D65kl','7TD6TIBQq^9>xDI','Lev@vehicula.net','2018-
04-05 04:20:55');
INSERT INTO "user" VALUES
('KUaIoNH0','QWF7ardrt)','Britanney@lacinia.net','201
8-04-06 04:04:40');
INSERT INTO "user" VALUES
('4cOwdNaJoV','#P.Ih!','Nita@viverra.gov','2018-04-13
04:01:23');
INSERT INTO "user" VALUES
('8MMeAQpXek','2CEMa','Patience@hendrerit.net','2018-
04-10 22:29:05');
INSERT INTO "user" VALUES
('4w98Clg','$q*','Cadman@Etiam.net','2018-04-12
00:24:45');
INSERT INTO "user" VALUES
('zzDv2','U^NW','Pascale@mollis.com','2018-04-18
18:12:44');
INSERT INTO "user" VALUES
('EUwjk3XZbO','5L/!uhxw>n6%d','Beau@fermentum.net','2
018-04-02 11:53:56');
INSERT INTO "user" VALUES ('Yux','J@rob^.-
&~^7N~P','Dean@ut.us','2018-04-01 10:07:04');
INSERT INTO "user" VALUES
('RAQG','Cv;&+/Os]^q','Timon@sit.edu','2018-04-12
23:00:20');
INSERT INTO "user" VALUES
('ljP7Hx',':U`nJfS}SKs','Celeste@iaculis.net','2018-
03-29 05:02:16');
INSERT INTO "user" VALUES ('r8kdC1G1M','Bp?
*y\xStf','Garth@libero.org','2018-03-28 16:30:10');
INSERT INTO "user" VALUES
('l1l6EX1AOF','M<K','Wallace@magnis.org','2018-04-08
15:40:44');
INSERT INTO "user" VALUES
('ENVtTNtN','YMc]Xx1ueWF','Lev@a.com','2018-04-02
20:56:43');
INSERT INTO "user" VALUES
('Azy6','K#*6DP?.nev=A\q','Aquila@eget.com','2018-03-
31 17:20:15');
INSERT INTO "user" VALUES
('cLU3QP','hKaO>fvJDc8@fon','Kennedy@nunc.edu','2018-
03-30 21:32:29');
INSERT INTO "user" VALUES
('sFHvz','mr5a7:S','Simone@justo.us','2018-03-28
01:23:47');
INSERT INTO "user" VALUES
('ZOVf','IAv}','Ferdinand@nunc.gov','2018-03-27
05:37:05');
INSERT INTO "user" VALUES
('Sj14g4X','5R`X:MW','Shelby@eget.com','2018-04-06
22:26:19');
INSERT INTO "user" VALUES
('SKUN','OUZWR5','Hyatt@In.gov','2018-03-31
02:37:13');
INSERT INTO "user" VALUES
```

```
('VH8G9Nc','Q(ZI','Marvin@montes.net','2018-04-16
21:24:49');
INSERT INTO "user" VALUES
('sk0jscnnsk','}tnV(u<Rg.','Fay@Aliquam.us','2018-04-
05 22:00:38');
INSERT INTO "user" VALUES
('Cvws','i#j=A>S9T','Joel@vehicula.net','2018-04-08
06:45:47');
INSERT INTO "user" VALUES ('jzqPr44d','kK0k}w?
_0gi','Cecilia@laoreet.net','2018-04-17 23:40:17');
INSERT INTO "user" VALUES
('fkP','7BY6wK7:H','Thaddeus@ac.edu','2018-03-27
21:04:43');
INSERT INTO "user" VALUES
('cnjiwB','4XWt','Phyllis@natoque.com','2018-03-27
07:43:51');
INSERT INTO "user" VALUES
('BlCiW','(E\}cRqyJ({6a]','Emily@urna.edu','2018-04-
11 00:21:55');
INSERT INTO "user" VALUES
('D2i','4P[WqD[j\~','Naida@Mauris.gov','2018-04-15
13:58:09');
INSERT INTO "user" VALUES
('tXE0UJ',')@nCjg','Macey@purus.org','2018-04-04
14:20:14');
INSERT INTO "user" VALUES ('EATrp','h~&?
4MFgwc1','Daniel@in.net','2018-03-28 04:43:39');
INSERT INTO "user" VALUES
('0vhmA5o2','dP._UKHIH','Olga@convallis.us','2018-04-
18 16:54:48');
INSERT INTO "user" VALUES
('RhmO','KB\^+PPixL:WP','Erica@eget.us','2018-04-14
01:03:13');
INSERT INTO "user" VALUES
('02t39yKwKo','acz0(!.s(cf)=','India@Class.net','2018
-04-18 10:45:45');
INSERT INTO "user" VALUES ('O9kk0','=iq}QlX(?
~;q#3','Brody@consectetuer.org','2018-04-15
21:23:56');
```

```
INSERT INTO "user" VALUES
('xavirt','$2y$10$3U/Uo5OTfiKohXC0f06TIu51gaGw8qFeaOR
3KRZ66GHC/WQYdKFm6','xfontes42@gmail.com','2018-04-06
18:30:24');
INSERT INTO "user" VALUES
('ana','$2y$10$3U/Uo5OTfiKohXC0f06TIu51gaGw8qFeaOR3KR
Z66GHC/WQYdKFm6','anaezes@gmail.com','2018-04-07
23:59:44');
INSERT INTO "user" VALUES
('mariana','$2y$10$3U/Uo5OTfiKohXC0f06TIu51gaGw8qFeaO
R3KRZ66GHC/WQYdKFm6','marianals@gmail.com','2018-04-
08 02:28:28');
INSERT INTO "user" VALUES
('eduardo','$2y$10$3U/Uo5OTfiKohXC0f06TIu51gaGw8qFeaO
R3KRZ66GHC/WQYdKFm6','edu.swimming@gmail.com','2018-
04-06 15:52:45');
INSERT INTO "user" VALUES
('jcl','$2y$10$3U/Uo5OTfiKohXC0f06TIu51gaGw8qFeaOR3KR
Z66GHC/WQYdKFm6','jcl@gmail.com','2018-04-04
01:47:20');

--CUSTOMER
INSERT INTO customer VALUES ('TYTTbQf','Harlan
Colon','12414  Argentina Ln.',16,'TRUE','FALSE');
INSERT INTO customer VALUES ('KXiAxRukz','Brenden
Sharp','69233 North Saint Helena
Ln.',74,'TRUE','FALSE');
INSERT INTO customer VALUES ('5zkyn6n5U','Hammett
Warner','27023  Anguilla St.',97,'FALSE','FALSE');
INSERT INTO customer VALUES ('CgsHHRf','Cullen
Espinoza','92547 South Anguilla
Way',32,'TRUE','FALSE');
INSERT INTO customer VALUES ('j5zBk','Dana
Jacobson','30838  Bangladesh
Blvd.',27,'TRUE','FALSE');
INSERT INTO customer VALUES ('0MM8g8','MacKenzie
Massey','17467 North Greenland
Ln.',37,'FALSE','FALSE');
INSERT INTO customer VALUES ('Ssuwu','Aiko
Kelley','95400 North Afghanistan
Blvd.',49,'FALSE','FALSE');
INSERT INTO customer VALUES ('PjujYy','Gwendolyn
Avery','52479  Chandler St.',83,'FALSE','FALSE');
INSERT INTO customer VALUES ('glbGcBfG','Cedric
Potter','2716 South Australia
Ct.',30,'FALSE','FALSE');
INSERT INTO customer VALUES ('l1mtgBKN','Jaden
Preston','53885 South Idaho Springs
Ln.',26,'FALSE','FALSE');
INSERT INTO customer VALUES ('NRb3xxj','Avram
Oneal','87753  Niger Ln.',91,'FALSE','FALSE');
INSERT INTO customer VALUES ('s53d41qTPk','Iris
Paul','29898 East Benin St.',22,'FALSE','FALSE');
INSERT INTO customer VALUES ('soMeMGrGpb','Brynn
Carr','56070  Italy Way',53,'FALSE','FALSE');
INSERT INTO customer VALUES ('UaUyY7HgDC','Neville
Stewart','98 East Cambodia Ct.',75,'FALSE','FALSE');
INSERT INTO customer VALUES ('x2CW','Dacey
Jarvis','32541 East Timor-leste
Ct.',69,'TRUE','FALSE');
INSERT INTO customer VALUES ('104zBsdD','Tara
Bullock','84542 East Council Bluffs
Ct.',79,'FALSE','FALSE');
INSERT INTO customer VALUES ('n1BRuWgdR','Quail
Burton','82990  Morocco Ln.',70,'TRUE','FALSE');
INSERT INTO customer VALUES ('D65kl','Asher
Combs','43205 South Botswana
Ave.',100,'FALSE','FALSE');
INSERT INTO customer VALUES ('KUaIoNH0','Carissa
Le','69139  Kyrgyzstan Ave.',77,'TRUE','FALSE');
INSERT INTO customer VALUES ('4cOwdNaJoV','Kane
Harding','32948 East Boulder Junction
Blvd.',42,'TRUE','FALSE');
INSERT INTO customer VALUES ('8MMeAQpXek','Quinn
Haney','66895 West Bosnia and Herzegovina
St.',49,'FALSE','FALSE');
INSERT INTO customer VALUES ('4w98Clg','Ginger
Fitzgerald','91487 East Derby
```

```
Ave.',82,'TRUE','FALSE');
INSERT INTO customer VALUES ('zzDv2','Karly
Faulkner','52765  Brazil Ave.',41,'FALSE','FALSE');
INSERT INTO customer VALUES ('EUwjk3XZbO','Damon
Oneil','56946 East Greenland
Way',24,'FALSE','FALSE');
INSERT INTO customer VALUES ('Yux','Velma
Bernard','62781 West Estonia
Blvd.',22,'FALSE','FALSE');
INSERT INTO customer VALUES ('RAQG','Baxter
Mcleod','5737 East Timor-leste
Blvd.',48,'TRUE','FALSE');
INSERT INTO customer VALUES ('ljP7Hx','Petra
Head','23636 North Fort Worth
Ave.',51,'FALSE','FALSE');
INSERT INTO customer VALUES ('r8kdC1G1M','Uriel
Alvarado','8969  Barrow Ave.',98,'TRUE','FALSE');
INSERT INTO customer VALUES ('l1l6EX1AOF','Dahlia
Harris','16319 South Peru St.',44,'FALSE','FALSE');
INSERT INTO customer VALUES ('ENVtTNtN','Signe
Clark','79440 East Darlington
Ave.',4,'FALSE','FALSE');
INSERT INTO customer VALUES ('Azy6','Aline
Francis','65673  Barbados St.',94,'TRUE','FALSE');
INSERT INTO customer VALUES ('cLU3QP','Nolan
Estrada','12783 North Macao
Blvd.',67,'FALSE','FALSE');
INSERT INTO customer VALUES ('sFHvz','Hedda
Baldwin','89319 South United Kingdom
St.',62,'FALSE','FALSE');
INSERT INTO customer VALUES ('ZOVf','Chester
Valenzuela','72011 South Spain
Ave.',31,'TRUE','FALSE');
INSERT INTO customer VALUES ('Sj14g4X','Keegan
Douglas','25247 East Greenland
Ln.',53,'TRUE','FALSE');
INSERT INTO customer VALUES ('SKUN','Ava Hobbs','1495
East Chad Way',49,'TRUE','FALSE');
INSERT INTO customer VALUES ('VH8G9Nc','Carson
Marks','38174 West Green River
Ln.',4,'TRUE','FALSE');
INSERT INTO customer VALUES ('sk0jscnnsk','Dennis
Sanchez','10688 East Aguadilla
Ct.',27,'TRUE','FALSE');
INSERT INTO customer VALUES ('Cvws','Quintessa
Vaughan','97425  Korea St.',53,'FALSE','FALSE');
INSERT INTO customer VALUES ('jzqPr44d','Veda
Bowman','77765  Mandan Way',42,'FALSE','FALSE');
INSERT INTO customer VALUES ('fkP','Cleo
Hopper','76754 South Laramie
Blvd.',5,'FALSE','FALSE');
INSERT INTO customer VALUES ('cnjiwB','Lucas
Hester','3704 East Berkeley Ln.',98,'TRUE','FALSE');
INSERT INTO customer VALUES ('BlCiW','Madeline
Oliver','44271 East Moldova Ln.',26,'TRUE','FALSE');
INSERT INTO customer VALUES ('D2i','Gage
Dodson','68511 West Palau Ave.',69,'FALSE','FALSE');
INSERT INTO customer VALUES ('tXE0UJ','Bruce
Ford','97384  Bosnia and Herzegovina
Ct.',53,'TRUE','FALSE');
INSERT INTO customer VALUES ('EATrp','Fitzgerald
Sheppard','14701  Bartlesville
Ct.',90,'FALSE','FALSE');
INSERT INTO customer VALUES ('0vhmA5o2','Lynn
Suarez','29366  Comoros Ct.',70,'TRUE','FALSE');
INSERT INTO customer VALUES ('RhmO','McKenzie
Maddox','92620 South Claremont
Blvd.',16,'TRUE','FALSE');
INSERT INTO customer VALUES ('02t39yKwKo','Joshua
Suarez','38582 North Gloucester
Ln.',45,'TRUE','FALSE');
INSERT INTO customer VALUES ('O9kk0','Marah
Johnston','9687 West Jamaica
Way',51,'FALSE','FALSE');

--MODERATOR
INSERT INTO moderator VALUES ('jcl');
```

```
--ADMINISTRATOR
INSERT INTO administrator VALUES ('xavirt');
INSERT INTO administrator VALUES ('ana');
INSERT INTO administrator VALUES ('mariana');
INSERT INTO administrator VALUES ('eduardo');

--BANNED
INSERT INTO banned VALUES ('TYTTbQf','2018-03-29
09:05:05','jcl');
INSERT INTO banned VALUES ('KXiAxRukz','2018-04-02
08:00:55','jcl');
INSERT INTO banned VALUES ('5zkyn6n5U','2018-03-30
14:04:26','jcl');
INSERT INTO banned VALUES ('CgsHHRf','2018-03-30
15:30:22','jcl');
INSERT INTO banned VALUES ('j5zBk','2018-04-01
23:05:45','jcl');

--CATEGORY
INSERT INTO category VALUES (1, 'Computers');
INSERT INTO category VALUES (2, 'Laptops');
INSERT INTO category VALUES (3, 'Mobile');
INSERT INTO category VALUES (4, 'Components');
INSERT INTO category VALUES (5, 'Storage');
INSERT INTO category VALUES (6, 'Peripherals');
INSERT INTO category VALUES (7, 'Photo');
INSERT INTO category VALUES (8, 'Video');
INSERT INTO category VALUES (9, 'Network');
INSERT INTO category VALUES (10, 'Software');

--PRODUCT
INSERT INTO product VALUES ('901896832','diam
blandit',5,7195.11,6.39,1.86,245);
INSERT INTO product VALUES ('672442768','dolor
taciti',5,3608.78,8.96,1.88,965);
INSERT INTO product VALUES
('186596482','Pellentesque',4,8215.91,4.96,4.66,800);
INSERT INTO product VALUES
('556397271','diam',6,3170.98,7.66,2.27,935);
```

```
INSERT INTO product VALUES ('696296971','Cras mattis
lectus',1,4329.14,8.12,1.19,454);
INSERT INTO product VALUES ('841341724','erat
nostra',9,3940.47,7.28,2.66,914);
INSERT INTO product VALUES
('780741675','Curabitur',2,4758.14,2.02,4.78,1026);
INSERT INTO product VALUES ('410469648','montes
pellentesque',7,8552.68,8.42,1.18,929);
INSERT INTO product VALUES ('901889798','elementum
suscipi',9,409.62,8.15,4.22,1138);
INSERT INTO product VALUES ('171093287','risus
sollicitudin',7,4699.83,6.70,3.75,451);
INSERT INTO product VALUES ('083108184','lacinia
fringilla',8,9117.49,5.60,4.93,964);
INSERT INTO product VALUES ('914510007','Curabitur
sociis dis nisi',2,4207.41,3.23,2.27,1342);
INSERT INTO product VALUES ('542291350','eleifend
Nulla',3,2673.58,5.02,3.78,1401);
INSERT INTO product VALUES ('891931229','tempus
sociis odio mattis magna',10,6118.55,8.32,2.59,630);
INSERT INTO product VALUES ('278925974','conubia
ornare lobortis',1,7101.92,7.27,1.02,290);

--COMMENT
INSERT INTO comment VALUES (1,'TYTTbQf', '2012-05-16
12:36:38', 'blakhinwuhdie hdie', 0, FALSE,
'901896832');
INSERT INTO comment VALUES (2,'KXiAxRukz', '2012-05-
16 11:39:38', 'blakdjhiwuhdie hdie', 0, FALSE,
'901896832');
INSERT INTO comment VALUES (3,'5zkyn6n5U', '2012-05-
16 13:36:38','blakahiwuhdie hdie', 0, FALSE,
'186596482');
INSERT INTO comment VALUES (4,'CgsHHRf', '2012-05-16
14:36:38','blakhkiwuhdie hdie', 0, FALSE,
'186596482');
INSERT INTO comment VALUES (5,'j5zBk', '2012-05-16
11:36:38','blakhiwujbhdie hdie', 0, FALSE,
'696296971');
```

```
INSERT INTO comment VALUES (6,'0MM8g8', '2012-05-16
17:36:38','blakhjiwuhdie hdie', 0, FALSE,
'696296971');
INSERT INTO comment VALUES (7,'Ssuwu', '2012-05-16
16:36:38','blakhijbwuhdie hdie', 0, FALSE,
'780741675');
INSERT INTO comment VALUES (8,'PjujYy', '2012-05-16
15:33:38','blakhijhbwuhdie hdie', 0, FALSE,
'780741675');
INSERT INTO comment VALUES (9,'glbGcBfG', '2012-05-16
15:32:38','blakhbjhiwuhdie hdie', 0, FALSE,
'171093287');
INSERT INTO comment VALUES (10,'l1mtgBKN', '2012-05-
16 15:26:38','blakhiwuhdie hdie', 0, FALSE,
'171093287');

--ANSWER
INSERT INTO answer VALUES (1,2);
INSERT INTO answer VALUES (3,4);
INSERT INTO answer VALUES (5,6);
INSERT INTO answer VALUES (7,8);
INSERT INTO answer VALUES (9,10);

--FLAGGED
INSERT INTO flagged VALUES (1, FALSE);
INSERT INTO flagged VALUES (2, FALSE);
INSERT INTO flagged VALUES (3, FALSE);
INSERT INTO flagged VALUES (4, FALSE);
INSERT INTO flagged VALUES (5, FALSE);
INSERT INTO flagged VALUES (6, FALSE);
INSERT INTO flagged VALUES (7, FALSE);
INSERT INTO flagged VALUES (8, FALSE);
INSERT INTO flagged VALUES (9, FALSE);
INSERT INTO flagged VALUES (10, FALSE);

--ATTRIBUTE
INSERT INTO attribute VALUES (1,'audio');
INSERT INTO attribute VALUES (2,'visual');
INSERT INTO attribute VALUES (3,'tatil');
INSERT INTO attribute VALUES (4,'alto');
INSERT INTO attribute VALUES (5,'comprido');

--ATTRIBUTE_PRODUCT
INSERT INTO attribute_product VALUES (1,'901896832',
'diam');
INSERT INTO attribute_product VALUES (2,'672442768',
'diam');
INSERT INTO attribute_product VALUES (3,'186596482',
'diam');
INSERT INTO attribute_product VALUES (4,'556397271',
'diam');
INSERT INTO attribute_product VALUES (5,'696296971',
'diam');
INSERT INTO attribute_product VALUES (1,'841341724',
'diam');
INSERT INTO attribute_product VALUES (2,'780741675',
'diam');
INSERT INTO attribute_product VALUES (3,'410469648',
'diam');
INSERT INTO attribute_product VALUES (4,'901889798',
'diam');
INSERT INTO attribute_product VALUES (5,'171093287',
'diam');
INSERT INTO attribute_product VALUES (1,'083108184',
'diam');
INSERT INTO attribute_product VALUES (2,'914510007',
'diam');
INSERT INTO attribute_product VALUES (3,'542291350',
'diam');
INSERT INTO attribute_product VALUES (4,'891931229',
'diam');
INSERT INTO attribute_product VALUES (5,'278925974',
'diam');

--CATEGORY_ATTRIBUTE
INSERT INTO category_attribute VALUES (1,1);
INSERT INTO category_attribute VALUES (2,2);
INSERT INTO category_attribute VALUES (3,3);
INSERT INTO category_attribute VALUES (4,4);
INSERT INTO category_attribute VALUES (5,4);
INSERT INTO category_attribute VALUES (1,6);
INSERT INTO category_attribute VALUES (2,7);
INSERT INTO category_attribute VALUES (3,8);
INSERT INTO category_attribute VALUES (4,9);
INSERT INTO category_attribute VALUES (4,10);

--FAVORITE
INSERT INTO favorite VALUES ('TYTTbQf','901896832');
INSERT INTO favorite VALUES
('KXiAxRukz','672442768');
INSERT INTO favorite VALUES
('5zkyn6n5U','186596482');
INSERT INTO favorite VALUES ('CgsHHRf','556397271');
INSERT INTO favorite VALUES ('j5zBk','696296971');
INSERT INTO favorite VALUES ('0MM8g8','841341724');
INSERT INTO favorite VALUES ('Ssuwu','780741675');
INSERT INTO favorite VALUES ('PjujYy','410469648');
INSERT INTO favorite VALUES ('glbGcBfG','901889798');
INSERT INTO favorite VALUES ('l1mtgBKN','171093287');

--PURCHASE
INSERT INTO purchase VALUES (1,'TYTTbQf', '2012-05-14
12:36:38', 899.02, 'Credit');
INSERT INTO purchase VALUES (2,'KXiAxRukz', '2012-05-
12 12:36:38', 899.02, 'Credit');
INSERT INTO purchase VALUES (3,'5zkyn6n5U', '2012-05-
11 12:36:38', 899.02,'Credit');
INSERT INTO purchase VALUES (4,'CgsHHRf', '2012-05-13
12:36:38', 899.02,'Credit');
INSERT INTO purchase VALUES (5,'j5zBk', '2012-05-10
12:36:38', 899.02,'Credit');
INSERT INTO purchase VALUES (6,'0MM8g8', '2012-05-09
12:36:38', 899.02,'Credit');
INSERT INTO purchase VALUES (7,'Ssuwu', '2012-05-08
12:36:38', 899.02,'Credit');
INSERT INTO purchase VALUES (8,'PjujYy', '2012-05-07
12:36:38', 899.02,'Debit');
```

```
INSERT INTO purchase VALUES (9,'glbGcBfG', '2012-05-
06 12:36:38', 899.02,'Debit');
INSERT INTO purchase VALUES (10,'l1mtgBKN', '2012-05-
05 12:36:38', 899.02,'Debit');
INSERT INTO purchase VALUES (11,'NRb3xxj', '2012-05-
04 12:36:38', 899.02,'Debit');
INSERT INTO purchase VALUES (12,'s53d41qTPk', '2012-
05-03 12:36:38', 899.02,'Debit');
INSERT INTO purchase VALUES (13,'soMeMGrGpb', '2012-
05-02 12:36:38', 899.02,'Paypal');
INSERT INTO purchase VALUES (14,'UaUyY7HgDC', '2012-
05-01 12:36:38', 899.02,'Paypal');
INSERT INTO purchase VALUES (15,'x2CW', '2012-05-16
12:36:38', 899.02,'Paypal');

--PURCHASE_PRODUCT
INSERT INTO purchase_product VALUES (1,'901896832',
899.02, 1);
INSERT INTO purchase_product VALUES (2,'672442768',
899.02, 1);
INSERT INTO purchase_product VALUES (3,'186596482',
899.02, 1);
INSERT INTO purchase_product VALUES (4,'556397271',
899.02, 1);
INSERT INTO purchase_product VALUES (5,'696296971',
899.02, 2);
INSERT INTO purchase_product VALUES (6,'841341724',
899.02, 1);
INSERT INTO purchase_product VALUES (7,'780741675',
899.02, 2);
INSERT INTO purchase_product VALUES (8,'410469648',
899.02, 1);
INSERT INTO purchase_product VALUES (9,'901889798',
899.02, 1);
INSERT INTO purchase_product VALUES (10,'171093287',
899.02, 1);
INSERT INTO purchase_product VALUES (11,'083108184',
899.02, 2);
INSERT INTO purchase_product VALUES (12,'914510007',
899.02, 1);
INSERT INTO purchase_product VALUES (13,'542291350',
899.02, 2);
INSERT INTO purchase_product VALUES (14,'891931229',
899.02, 1);
INSERT INTO purchase_product VALUES (15,'278925974',
899.02, 1);

--RATING
INSERT INTO rating VALUES ('TYTTbQf','901896832', 5);
INSERT INTO rating VALUES ('KXiAxRukz','672442768',
1);
INSERT INTO rating VALUES ('5zkyn6n5U','186596482',
2);
INSERT INTO rating VALUES ('CgsHHRf','186596482', 3);
INSERT INTO rating VALUES ('j5zBk','186596482', 3);
INSERT INTO rating VALUES ('0MM8g8','780741675', 3);
INSERT INTO rating VALUES ('Ssuwu','410469648', 3);
INSERT INTO rating VALUES ('PjujYy','780741675', 3);
INSERT INTO rating VALUES ('glbGcBfG','410469648',
1);
INSERT INTO rating VALUES ('l1mtgBKN','171093287',
4);
INSERT INTO rating VALUES ('NRb3xxj','083108184', 4);
INSERT INTO rating VALUES ('s53d41qTPk','914510007',
4);
INSERT INTO rating VALUES ('soMeMGrGpb','542291350',
4);
INSERT INTO rating VALUES ('UaUyY7HgDC','891931229',
5);
INSERT INTO rating VALUES ('x2CW','278925974', 5);
```

Revision history

- Added a small introduction.
- Changed magnitude and estimated growth of users/customers.

- Fixed SELECT02.
- Added SELECT08 and edited SELECT03 to reflect Full Text Search functionality.
- Changed IDX04 and added IDX05 to reflect Full Text Search functionality.
- Added TRIGGER05 and TRIGGER06 to reflect Full Text Search functionality.
- Added more justification on indexes.

GROUP1723, 11/04/2018

Ana Cláudia Fonseca Santos, up200700742@fe.up.pt

Eduardo de Mendonça Rodrigues Salgado Ramos, up201505779@fe.up.pt

Mariana Lopes da Silva, up201506197@fe.up.pt

Xavier Reis Fontes, up201503145@fe.up.pt