A6: Indexes, triggers, user functions, transactions and population

The product consists of an online auction website where you are able to bid on and create auctions.

In this artifact, we intend to define the physical schema of our database. We will present a prediction of the database load, create indexes to improve performance and create triggers to assure the integrity of the database whenever certain values in certain tables are inserted, updated or deleted.

1. Database Workload

After defining the physical structure of a database, it is extremely important to study its workload. The study of a database workload consists in two main areas: the estimation of how many tuples will exist for each relation and the identification of the most frequent queries and modifications.

1.1. Tuple Estimation

Relation reference	Relation Name	Order of magnitude	Estimated growth
R01	auction	tens of thousands	dozens per day
R02	bids	hundreds of thousands	hundreds per day
R03	notification	hundreds of thousands	hundreds per day
R04	user	thousands	dozens per day
R05	admin	unit	no growth
R06	buyer	thousands	dozens per day
R07	seller	thousands	dozens per day
R08	block	dozens	units per day
R09	shipping_method	units	no growth
R10	ships	tens of thousands	dozens per day
R11	payment_method	hundreds of thousands	dozens per day
R12	accepts	tens of thousands	dozens per day
R13	reports	tens of thousands	units per day
R14	report_status	units	no growth
R15	watchlists	hundreds of thousands	hundreds per day
R16	skill	tens	no growth
R17	features	hundreds of thousands	hundreds per day
R18	main_color	tens of thousands	dozens per day

Relation reference	Relation Name	Order of magnitude	Estimated growth
R19	development_stage	tens of thousands	dozens per day
R20	category	tens of thousands	dozens per day
R21	auction_status	units	no growth
R22	image	tens of thousands	dozens per day
R23	profile_photo	thousands	dozens per day
R24	animal_photo	hundreds of thousands	hundreds per day

1.2. Frequent Queries

Query	SELECT01
Description	View User's Profile
Frequency	hundreds per day

```
SELECT name, email, url
    FROM ("user" LEFT JOIN (profile_photo NATURAL JOIN "image") ON "user".id =
profile_photo.id_user)
    WHERE "user".id = $id_user;
```

Query	SELECT02
Description	Search for Auctions with Filters
Frequency	thousands per day

```
SELECT DISTINCT auction.id, species_name, current_price, age, ending_date
    FROM (auction JOIN features ON auction.id = features.id_auction)
    WHERE   (id_category = $category OR $category IS NULL)
        AND (id_main_color = $main_color OR $main_color IS NULL)
        AND (id_dev_stage = $dev_stage OR $dev_stage IS NULL)
        AND (current_price < $max_price OR $max_price IS NULL)
        AND (id_skill IN ($climbs, $jumps, $talks, $skates, $olfaction,
$navigation, $echo, $acrobatics))
        AND id_status = 0
    ORDER BY ending_date;
```

Query	SELECT03
Description	Full Text Search for Auction

Query	SELECT03
Frequency	thousands per day

```
SELECT auction.id, species_name, current_price, age, ending_date,
ts_rank_cd(textsearch, query) AS rank
    FROM auction, to_tsquery($search) AS query,
        to_tsvector(name || ' ' || species_name || ' ' || description ) AS
textsearch
    WHERE query @@ textsearch AND id_status = 0
    ORDER BY rank DESC;
```

Query	SELECT04
Description	Full Text Search for Users in Admin Dashboard
Frequency	dozens per day

```
SELECT "name", email, ts_rank_cd(textsearch, query) AS rank
    FROM "user", to_tsquery($search) AS query,
        to_tsvector(name || ' ' || email) AS textsearch
    WHERE query @@ textsearch
    ORDER BY rank DESC;
```

Query	SELECT05
Description	Top 3 Auctions for Homepage Display
Frequency	hundreds per day

```
SELECT auctions.id, auctions.species_name, auctions.current_price, auctions.age,
auctions.ending_date
    FROM (SELECT auction.*, count(*) AS num_bids
            FROM  (auction JOIN bids ON auction.id = bids.id_auction)
            WHERE id_status = 0
            GROUP BY  auction.id) AS auctions
    ORDER BY num_bids DESC
    LIMIT 3;
```

Query	SELECT06
Description	View Notifications

Query	SELECT06
Frequency	thousands per day

```
SELECT *
    FROM "notification"
    WHERE id_buyer = $id_buyer
    ORDER BY id DESC
    LIMIT 5;
```

Query	SELECT07
Description	View Auction Details
Frequency	thousands per day

```
SELECT auction.*, skill.TYPE, category.TYPE, main_color.TYPE,
development_stage.TYPE
    FROM ((((auction JOIN features ON auction.id = features.id_auction)
            JOIN skill ON skill.id = features.id_skill)
            JOIN category ON category.id = auction.id_category)
            JOIN main_color ON main_color.id = auction.id_main_color)
            JOIN development_stage ON development_stage.id = auction.id_dev_stage
    WHERE auction.id = $id_auction;
```

Query	SELECT08
Description	View Watchlisted Auctions
Frequency	hundreds per day

```
SELECT auction.id, species_name, current_price, age, ending_date
    FROM (watchlists JOIN auction ON auction.id = watchlists.id_auction)
    WHERE watchlists.id_buyer = $id_buyer AND id_status = 0
    ORDER BY ending_date;
```

Query	SELECT09
Description	View My Auctions as a Seller
Frequency	hundreds per day

```
SELECT auction.id, species_name, current_price, age, ending_date
    FROM auction
    WHERE auction.id_seller = $id_seller AND id_status = 0
    ORDER BY ending_date;
```

Query	SELECT10
Description	View Purchase History
Frequency	hundreds per day

```
SELECT auction.id, species_name, current_price, age, ending_date
    FROM auction
    WHERE auction.id_winner = $id_buyer AND id_status = 1
    ORDER BY ending_date DESC;
```

Query	SELECT11
Description	View Bidded Ongoing Auctions
Frequency	hundreds per day

```
SELECT DISTINCT auction.id, species_name, current_price, age, ending_date
    FROM auction JOIN bids ON bids.id_auction = auction.id
    WHERE bids.id_buyer = $id_buyer AND id_status = 0
    ORDER BY ending_date;
```

Query	SELECT12
Description	View Last 5 Bids in an Auction
Frequency	thousands per day

```
SELECT bids.value, "user".name
    FROM (bids JOIN "user" ON "user".id = id_buyer)
    WHERE bids.id_auction = $id_auction
    ORDER BY bids.id DESC
    LIMIT 5;
```

1.3. Frequent Updates

Query	INSERT01
Description	Create a New User
Frequency	dozens per day

```
INSERT INTO "user" (name, email, hashed_password)
    values ($name, $email, $hashed_password);
```

Query	INSERT02
Description	Create a New Auction
Frequency	dozens per day

```
INSERT INTO auction (name, description, species_name, age, starting_price,
buyout_price, current_price, ending_date, rating_seller, id_category,
id_main_color, id_dev_stage, id_seller, id_status)
    values ($name, $description, $species_name, $age, $starting_price,
$buyout_price, $current_price, $ending_date, $rating_seller, $id_category,
$id_main_color, $id_dev_stage, $id_seller, $id_status);
```

Query	INSERT03
Description	Create a New Bid
Frequency	hundreds per day

```
INSERT INTO bids(value, maximum, id_auction, id_buyer)
    values ($value, $maximum, $id_auction, $id_buyer );
```

Query	INSERT04
Description	Create a New Notification
Frequency	hundreds per day

```
INSERT INTO "notification" ("message", "type", "read", id_auction, id_buyer)
    values ($message, $type, false, $id_auction, $id_buyer);
```

Query	INSERT05
Description	Block a new User
Frequency	units per day

```
INSERT INTO blocks (end_date, id_admin, id_seller)
    values($end_date, $id_admin, $id_seller);
```

Query	INSERT06
Description	Create a New Report
Frequency	units per day

```
INSERT INTO reports("date", id_buyer, id_seller, id_status)
    values($date, $id_buyer, $id_seller, $id_status);
```

Query	INSERT07
Description	Add an Auction to the Watchlist
Frequency	hundreds per day

```
INSERT INTO watchlists (id_auction, id_buyer)
    values($id_auction, id_buyer);
```

Query	INSERT08
Description	Add a Status to a Report
Frequency	units per day

```
INSERT INTO report_status (id, TYPE)
    values($id, $report_status);
```

Query	INSERT09
Description	Add a Status to an Auction

Query	INSERT09
Frequency	dozens per day

```
INSERT INTO auction_status (id, TYPE)
    values($id, $auction_status);
```

Query	UPDATE01
Description	Edit User's Profile
Frequency	units per day

```
UPDATE "user"
    SET name=$name, email=$email, hashed_password = $password
    WHERE id = $id_user
```

Query	UPDATE02
Description	Edit an Auction
Frequency	units per day

```
UPDATE auction
    SET name=$name, description=$description, species_name=$species_name,
age=$age, starting_price = $starting_price, buyout_price = $buyout_price,
ending_date=$ending_date, id_category = $id_category, id_main_color =
$id_main_color, id_dev_stage = $id_dev_stage, id_status = $id_status
    WHERE id = $id_auction AND id_seller = $id_seller
```

Query	UPDATE03
Description	Mark a Notification as read
Frequency	dozens per day

```
UPDATE "notification"
    SET "read" = $read
    WHERE id_auction = $id_auction AND id_buyer = $id_buyer;
```

Query	DELETE01
Description	Delete an Account
Frequency	units per day

```
DELETE FROM "user"
    WHERE id = $id_user;
```

Query	DELETE02
Description	Delete an Auction
Frequency	units per day

```
DELETE FROM auction
    WHERE id = $id_auction;
```

Query	DELETE03
Description	Remove Auction from Watchlist
Frequency	dozens per day

```
DELETE FROM watchlists
    WHERE id_auction = $id_auction AND id_buyer = $id_buyer;
```

2. Proposed Indices

After identifying the most frequent queries to the database, it is important to analyze how we can improve the results in order to obtain higher performances and more relevant results. In order to do this, we propose the indexes mentioned below.

2.1. Performance Indices

Index	IDX01
Related queries	SELECT06
Relation	notification

Index	IDX01
Attribute	id_buyer
Type	Hash
Cardinality	medium
Clustering	Yes
Justification	Given that this query is executed many times, it's performance is extremely important. An hashed index will allow a linear search through the table. The id_buyer attribute has medium cardinality due to the fact that an user can have multiple notifications associated to him. For this same reason, this index is a good candidate for clustering.

```
CREATE INDEX notification_id ON "notifications" USING hash(id_buyer);
```

Index	IDX02
Related queries	SELECT08
Relation	watchlists
Attribute	id_buyer
Type	Hash
Cardinality	medium
Clustering	Yes
Justification	Given that this query is executed many times, it's performance is extremely important. An hashed index will allow a linear search through the table. The id_buyer attribute has medium cardinality due to the fact that an user can have multiple auction in his watchlist. For this same reason, this index is a good candidate for clustering.

```
CREATE INDEX watchlists_id ON watchlists USING hash(id_buyer);
```

Index	IDX03
Related queries	SELECT09
Relation	auction
Attribute	id_seller

Index	IDX03
Type	B-tree
Cardinality	medium
Clustering	No
Justification	Given that this query is executed many times, it's performance is extremely important. It will help search for an user's auction and order them by ending date faster. The id_seller attribute has medium cardinality due to the fact that a seller can create multiple auctions. It's clustered to allow for quick range queries.

```
CREATE INDEX auction_id ON auction USING btree(id_seller);
```

Index	IDX04
Related queries	SELECT11
Relation	bids
Attribute	id_buyer
Type	Hash
Cardinality	medium
Clustering	Yes
Justification	Given that this query is executed many times, it's performance is extremely important. An hashed index will allow a linear search through the table. The id_buyer attribute has medium cardinality due to the fact that a buyer can bid multiple times and in multiple auctions. For this same reason, this index is a good candidate for clustering.

```
CREATE INDEX bids_id ON bids USING hash(id_buyer);
```

2.2. Full-text Search Indices

Index	IDX01
Related queries	SELECT03
Relation	auction
Attribute	name

Index	**IDX01**
Type	GiST
Clustering	No
Justification	The index type choosen was GiST because it is better than GIN to deal with dynamic data. This index is proposed in order to improve the performance of full text searches while searching for auctions.

```
CREATE INDEX search_auction ON auction USING GIST (to_tsvector('english', name ||
' ' || species_name || ' ' || description ))
```

Index	**IDX02**
Related queries	SELECT04
Relation	user
Attribute	name
Type	GiST
Clustering	No
Justification	The index type choosen was GiST because it is better than GIN to deal with dynamic data. This index is proposed in order to improve the performance of full text searches while searching for users in the admin dashboard.

```
CREATE INDEX admin_search ON "user" USING GIST (to_tsvector('english', name || ' '
|| email ))
```

3. Triggers

Trigger	**TRIGGER01**
Description	Create Buyer whenever a new User is created

```
CREATE FUNCTION create_buyer() RETURNS TRIGGER AS
$BODY$
BEGIN
    INSERT INTO buyer(id) VALUES (NEW.id);
    RETURN NULL;
END
```

```
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER create_buyer
    AFTER INSERT ON "user"
    FOR EACH ROW
    EXECUTE PROCEDURE create_buyer();
```

Trigger	**TRIGGER02**
Description	Create Seller whenever an user creates his first auction

```
CREATE FUNCTION create_seller() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF NOT EXISTS(SELECT * FROM auction WHERE NEW.id = id)
       THEN INSERT INTO seller(id) values (NEW.id) ;
    END IF;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER create_seller
    BEFORE INSERT ON auction
    FOR EACH ROW
    EXECUTE PROCEDURE create_seller();
```

Trigger	**TRIGGER03**
Description	Stop all ongoing auctions when a seller is blocked

```
CREATE FUNCTION stop_ongoing_auctions() RETURNS TRIGGER AS
$BODY$
BEGIN
    UPDATE auction
        SET id_status = 2
        WHERE id IN (SELECT id FROM auction WHERE id_seller = NEW.id_seller) AND
id_status = 0;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER stop_ongoing_auctions
    AFTER INSERT ON blocks
```

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

Trigger	TRIGGER04
Description	Update current price on auction after a bid

```
CREATE FUNCTION update_current_price() RETURNS TRIGGER AS
$BODY$
BEGIN
    UPDATE auction
        SET current_price = NEW.value
        WHERE id = NEW.id_auction;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER update_current_price
    AFTER INSERT ON bids
    FOR EACH ROW
    EXECUTE PROCEDURE update_current_price();
```

Trigger	TRIGGER05
Description	Send Notification to buyers when their bid is surpassed

```
CREATE FUNCTION send_notification() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF EXISTS(SELECT * FROM bids WHERE id_auction = NEW.id_auction)
    THEN
        INSERT INTO "notification" ("message", "read", id_auction, id_buyer)
        SELECT 'Your bid has been surpassed', FALSE, info.id_auction,
info.id_buyer
        FROM (SELECT value, id_auction, id_buyer
                FROM  bids
                WHERE id_auction = NEW.id_auction
                ORDER BY value DESC
                LIMIT 1
            ) AS info;
    END IF;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;
```

```
CREATE TRIGGER send_notification
    BEFORE INSERT ON bids
    FOR EACH ROW
    EXECUTE PROCEDURE send_notification();
```

Trigger	TRIGGER06
Description	Send Notification to winner when the status changes to finished

```
CREATE FUNCTION notify_winner() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF(NEW.id_status <> OLD.id_status)
    THEN
        INSERT INTO "notification" ("message", "read", id_auction, id_buyer)
        SELECT 'You won an auction!', FALSE, info.id_auction, info.id_buyer
        FROM (SELECT value, id_auction, id_buyer
                FROM  bids
                WHERE id_auction = NEW.id
                ORDER BY value DESC
                LIMIT 1
            ) AS info;
    END IF;
    RETURN NULL;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER notify_winner
    AFTER UPDATE ON auction
    FOR EACH ROW
    EXECUTE PROCEDURE notify_winner();
```

Trigger	TRIGGER07
Description	Verify if the value of a new bid is greater than the current one

```
CREATE FUNCTION verify_bid_value() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF EXISTS (SELECT * FROM auction WHERE NEW.id_auction = id AND current_price >
NEW.value )
        THEN RAISE EXCEPTION 'A bid can only be made with a value greater than the
current bid';
    END IF;
    RETURN NEW;
END
```

```
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER verify_bid_value
    BEFORE INSERT ON bids
    FOR EACH ROW
    EXECUTE PROCEDURE verify_bid_value();
```

Trigger	TRIGGER08
Description	Update seller's rating when a rating is added

```
CREATE FUNCTION update_rating() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF (NEW.rating_seller >= 1 AND NEW.rating_seller <= 5)
    THEN
        UPDATE seller
        SET rating = (
            SELECT AVG(rating_seller)
            FROM auction
            WHERE auction.id = NEW.id AND auction.rating_seller >= 1 AND
auction.rating_seller <= 5
        )
        WHERE seller.id = NEW.id_seller;
    END IF;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;

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

Trigger	TRIGGER09
Description	Remove auction from watchlists after it finishes

```
CREATE FUNCTION remove_watchlists() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF EXISTS(SELECT * FROM auction WHERE id_status = NEW.id_status AND
NEW.id_status = 1)
    THEN
        DELETE FROM watchlists
```

```
        WHERE id_buyer = (SELECT id_buyer FROM watchlists WHERE
watchlists.id_auction = NEW.id);
    END IF;
    RETURN NULL;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER remove_watchlists
    AFTER UPDATE ON auction
    FOR EACH ROW
    EXECUTE PROCEDURE remove_watchlists();
```

4. Transactions

Transactions are used to assure the integrity of the database by defining isolation levels that allow keeping the data stable throughout the transaction or defining behavior when something fails.

SQL Reference	**Select Reports to display in Admin Dashboard**
Justification	When viewing the administrator dashboard, there are two queries that need to occur to get all the necessary data of the reports: the number of reports to be able to define the pagination and the 10 most recent reports to be displayed in the first page. Therefore, this transaction assures that the number of pages is consistent with the number of reports, even when a report is added between the two queries. It's READ ONLY because it only uses Selects.
Isolation level	SERIALIZABLE READ ONLY

```
BEGIN TRANSACTION ISOLATION LEVEL SERIALIZABLE READ ONLY;

SELECT COUNT(*)
FROM Reports;

SELECT reports.date, report_status.TYPE, B.name, "user".name
FROM ((((SELECT "user".name AS name, buyer.id AS id FROM buyer JOIN "user" ON
"user".id = buyer.id) AS B
    JOIN reports ON reports.id_buyer = B.id) JOIN report_status ON
report_status.id = reports.id_status)
    JOIN seller ON reports.id_seller = seller.id JOIN "user" ON "user".id =
seller.id)
ORDER BY reports.date DESC
LIMIT 10;

COMMIT;
```

SQL Reference	Select Auction Information and Bidding History
Justification	When viewing an auction, there are two queries that need to occur to get all the data related to the auction: the auction information (species name, description, current price, etc) and the bidding history (last 5 bids). Since the last bid corresponds to the current price, then this these two values must be equal. Therefore, this transaction assures that these values match even when a new bid occurs between the two queries. It's READ ONLY because it only uses Selects.
Isolation level	SERIALIZABLE READ ONLY

```
BEGIN TRANSACTION;
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE READ ONLY;

SELECT auction.*, skill.TYPE, category.TYPE, main_color.TYPE,
development_stage.TYPE
    FROM ((((auction JOIN features ON auction.id = features.id_auction)
            JOIN skill ON skill.id = features.id_skill)
            JOIN category ON category.id = auction.id_category)
            JOIN main_color ON main_color.id = auction.id_main_color)
            JOIN development_stage ON development_stage.id = auction.id_dev_stage
    WHERE auction.id = $id_auction;

SELECT bids.value, "user".name
    FROM (bids JOIN "user" ON "user".id = id_buyer)
    WHERE bids.id_auction = $id_auction
    ORDER BY bids.id DESC
    LIMIT 5;

COMMIT;
```

SQL Reference	Select Auctions in Profile
Justification	When a buyer views its profile, he has access to its purchase history (past auctions he won) and its ongoing auctions (ongoing auctions that he has bidded). If the ending date of one of the ongoing auctions is reached between the two queries and the buyer wins, that auction should be displayed in the purchase history and not in the ongoing auctions. Therefore, this transaction assures that the status of the auction is the same in these two queries. It's READ ONLY because it only uses Selects.
Isolation level	SERIALIZABLE READ ONLY

```
BEGIN TRANSACTION;
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE READ ONLY;

SELECT auction.id, species_name, current_price, age, ending_date
    FROM auction
    WHERE auction.id_winner = $id_buyer AND id_status = 1
    ORDER BY ending_date DESC;

SELECT DISTINCT auction.id, species_name, current_price, age, ending_date
    FROM auction JOIN bids ON bids.id_auction = auction.id
    WHERE bids.id_buyer = $id_buyer AND id_status = 1
    ORDER BY ending_date;

COMMIT;
```

5. SQL Code

5.1. Database schema

```
 DROP TABLE IF EXISTS watchlists;
DROP TABLE IF EXISTS ships;
DROP TABLE IF EXISTS profile_photo;
DROP TABLE IF EXISTS features;
DROP TABLE IF EXISTS animal_photo;
DROP TABLE IF EXISTS accepts;
DROP TABLE IF EXISTS skill;
DROP TABLE IF EXISTS reports;
DROP TABLE IF EXISTS report_status;
DROP TABLE IF EXISTS "notification";
DROP TABLE IF EXISTS bids;
DROP TABLE IF EXISTS auction;
DROP TABLE IF EXISTS auction_status;
DROP TABLE IF EXISTS main_color;
DROP TABLE IF EXISTS "image";
DROP TABLE IF EXISTS blocks;
DROP TABLE IF EXISTS development_stage;
DROP TABLE IF EXISTS category;
DROP TABLE IF EXISTS shipping_method;
DROP TABLE IF EXISTS payment_method;
DROP TABLE IF EXISTS "admin";
DROP TABLE IF EXISTS seller;
DROP TABLE IF EXISTS buyer;
DROP TABLE IF EXISTS "user";

DROP TYPE IF EXISTS skill_name;
DROP TYPE IF EXISTS category_name;
DROP TYPE IF EXISTS shipping;
DROP TYPE IF EXISTS payment;
```

```
DROP TYPE IF EXISTS dev_stage;
DROP TYPE IF EXISTS color;
DROP TYPE IF EXISTS report_status_name;
DROP TYPE IF EXISTS auction_status_name;

DROP INDEX IF EXISTS user_email;
DROP INDEX IF EXISTS search_auction;
DROP INDEX IF EXISTS admin_search;
DROP INDEX IF EXISTS auction_id;
DROP INDEX IF EXISTS notification_id;

DROP TRIGGER IF EXISTS create_buyer ON "user";
DROP TRIGGER IF EXISTS create_seller ON "user";
DROP TRIGGER IF EXISTS stop_ongoing_auctions ON blocks;
DROP TRIGGER IF EXISTS update_current_price ON bids;
DROP TRIGGER IF EXISTS send_notification ON bids;
DROP TRIGGER IF EXISTS notify_winner ON auction;
DROP TRIGGER IF EXISTS verify_bid_value ON bids;
DROP TRIGGER IF EXISTS update_rating ON auction;
DROP TRIGGER IF EXISTS remove_watchlists ON auction;

DROP FUNCTION IF EXISTS create_buyer();
DROP FUNCTION IF EXISTS create_seller();
DROP FUNCTION IF EXISTS stop_ongoing_auctions();
DROP FUNCTION IF EXISTS update_current_price();
DROP FUNCTION IF EXISTS send_notification();
DROP FUNCTION IF EXISTS notify_winner();
DROP FUNCTION IF EXISTS verify_bid_value();
DROP FUNCTION IF EXISTS update_rating();
DROP FUNCTION IF EXISTS remove_watchlists();
----------------------------------------
-- TYPES
----------------------------------------
CREATE TYPE shipping AS ENUM ('Standard Mail', 'Express Mail', 'Urgent Mail');
CREATE TYPE payment AS ENUM ('Debit Card', 'PayPal');
CREATE TYPE skill_name AS ENUM ('Climbs', 'Jumps', 'Talks', 'Skates', 'Olfaction',
'Moonlight Navigation', 'Echolocation', 'Acrobatics');
CREATE TYPE color AS ENUM ('Blue', 'Brown', 'Black', 'Yellow', 'Green', 'Red',
'White');
CREATE TYPE dev_stage AS ENUM ('Baby', 'Child', 'Teen', 'Adult', 'Elderly');
CREATE TYPE category_name AS ENUM ('Mammals', 'Insects', 'Reptiles', 'Fishes',
'Birds', 'Amphibians');
CREATE TYPE report_status_name as ENUM('Pending', 'Approved', 'Denied');
CREATE TYPE auction_status_name as ENUM('Ongoing', 'Finished', 'Cancelled');

----------------------------------------
-- TABLES
----------------------------------------
CREATE TABLE "user"
(
    id SERIAL PRIMARY KEY,
    name text NOT NULL,
    email text NOT NULL UNIQUE,
    hashed_password text NOT NULL
```

```
);

CREATE TABLE "admin"
(
    id integer NOT NULL PRIMARY KEY REFERENCES "user" (id) ON UPDATE CASCADE ON
DELETE RESTRICT
);

CREATE TABLE buyer
(
    id integer NOT NULL PRIMARY KEY REFERENCES "user" (id) ON UPDATE CASCADE ON
DELETE CASCADE
);

CREATE TABLE seller
(
    id integer NOT NULL PRIMARY KEY REFERENCES "user" (id) ON UPDATE CASCADE ON
DELETE CASCADE,
    rating NUMERIC(3, 2) CHECK (rating >= 1 AND rating <= 5)
);

CREATE TABLE skill
(
    id SERIAL PRIMARY KEY,
    TYPE skill_name NOT NULL
);

CREATE TABLE main_color
(
    id SERIAL PRIMARY KEY,
    TYPE color NOT NULL
);

CREATE TABLE development_stage
(
    id SERIAL PRIMARY KEY,
    TYPE dev_stage NOT NULL
);

CREATE TABLE category
(
    id SERIAL PRIMARY KEY,
    TYPE category_name NOT NULL
);

CREATE TABLE payment_method
(
    id SERIAL PRIMARY KEY,
    TYPE payment NOT NULL
);

CREATE TABLE shipping_method
(
    id SERIAL PRIMARY KEY,
```

```
    TYPE shipping NOT NULL
);

CREATE TABLE auction_status
(
    id integer PRIMARY KEY,
    TYPE auction_status_name NOT NULL
);

CREATE TABLE auction
(
    id SERIAL PRIMARY KEY,
    name text NOT NULL,
    description text NOT NULL,
    species_name text NOT NULL,
    age integer NOT NULL,
    starting_price integer NOT NULL,
    buyout_price integer,
    current_price integer,
    ending_date date NOT NULL,
    rating_seller integer CHECK (rating_seller >= 1 AND rating_seller <= 5)
DEFAULT NULL,
    id_category integer NOT NULL REFERENCES category (id) ON UPDATE CASCADE ON
DELETE RESTRICT,
    id_main_color integer NOT NULL REFERENCES main_color (id) ON UPDATE CASCADE ON
DELETE RESTRICT,
    id_dev_stage integer NOT NULL REFERENCES development_stage (id) ON UPDATE
CASCADE ON DELETE RESTRICT,
    id_payment_method integer REFERENCES payment_method (id) ON UPDATE CASCADE ON
DELETE RESTRICT,
    id_shipping_method integer REFERENCES shipping_method (id) ON UPDATE CASCADE
ON DELETE RESTRICT,
    id_seller integer NOT NULL REFERENCES seller (id) ON UPDATE CASCADE ,
    id_winner integer REFERENCES buyer (id) ON UPDATE CASCADE ,
    id_status integer NOT NULL REFERENCES auction_status (id) ON UPDATE CASCADE,
    CONSTRAINT "buyout_price_ck" CHECK (buyout_price > starting_price),
    CONSTRAINT "current_price_ck" CHECK (current_price >= starting_price),
    CONSTRAINT "ending_date_ck" CHECK ((ending_date > 'now'::text::date) OR
(id_status = 1 OR id_status = 2))
);

CREATE TABLE bids
(
    id SERIAL PRIMARY KEY,
    value integer NOT NULL,
    maximum integer,
    id_auction integer NOT NULL REFERENCES auction (id) ON UPDATE CASCADE ON
DELETE CASCADE,
    id_buyer integer REFERENCES buyer (id) ON UPDATE CASCADE,
    CONSTRAINT "maximum_ck" CHECK (maximum >= value)
);

CREATE TABLE "notification"
(
```

```
    id SERIAL PRIMARY KEY,
    "message" text NOT NULL,
    "read" boolean DEFAULT FALSE,
    id_auction integer NOT NULL REFERENCES auction (id) ON UPDATE CASCADE ON
DELETE CASCADE,
    id_buyer integer REFERENCES buyer (id) ON UPDATE CASCADE ON DELETE CASCADE
);

CREATE TABLE blocks
(
    id SERIAL PRIMARY KEY,
    end_date date NOT NULL CHECK (end_date > 'now'::text::date),
    id_admin integer NOT NULL REFERENCES "admin" (id) ON UPDATE CASCADE,
    id_seller integer NOT NULL REFERENCES seller (id) ON UPDATE CASCADE ON DELETE
CASCADE
);

CREATE TABLE ships
(
    id_seller integer NOT NULL REFERENCES seller (id) ON UPDATE CASCADE ON DELETE
CASCADE,
    id_shipping_method integer NOT NULL REFERENCES shipping_method (id) ON UPDATE
CASCADE ON DELETE CASCADE,
    PRIMARY Key(id_seller, id_shipping_method)
);

CREATE TABLE accepts
(
    id_seller integer NOT NULL REFERENCES seller (id) ON UPDATE CASCADE ON DELETE
CASCADE,
    id_payment_method integer NOT NULL REFERENCES payment_method (id) ON UPDATE
CASCADE ON DELETE CASCADE,
    PRIMARY Key(id_seller, id_payment_method)
);

CREATE TABLE report_status
(
    id integer PRIMARY KEY,
    TYPE report_status_name NOT NULL
);

CREATE TABLE reports
(
    id SERIAL PRIMARY KEY,
    "date" date NOT NULL DEFAULT 'now'::text::date,
    id_buyer integer NOT NULL REFERENCES buyer (id) ON UPDATE CASCADE,
    id_seller integer NOT NULL REFERENCES seller (id) ON UPDATE CASCADE ON DELETE
CASCADE,
    id_status integer NOT NULL REFERENCES report_status ON UPDATE CASCADE
);

CREATE TABLE watchlists
(
    id_auction integer NOT NULL REFERENCES auction (id) ON UPDATE CASCADE ON
```

```
DELETE CASCADE,
    id_buyer integer NOT NULL REFERENCES buyer (id) ON UPDATE CASCADE ON DELETE
CASCADE,
    PRIMARY Key(id_auction, id_buyer)
);

CREATE TABLE features
(
    id_auction integer NOT NULL REFERENCES auction (id) ON UPDATE CASCADE ON
DELETE CASCADE,
    id_skill integer NOT NULL REFERENCES skill (id) ON UPDATE CASCADE ON DELETE
CASCADE,
    PRIMARY Key(id_auction, id_skill)
);

CREATE TABLE "image"
(
    id SERIAL PRIMARY KEY,
    url text NOT NULL
);

CREATE TABLE profile_photo
(
    id integer NOT NULL PRIMARY KEY REFERENCES "image" (id) ON UPDATE CASCADE,
    id_user integer NOT NULL REFERENCES "user" (id) ON UPDATE CASCADE ON DELETE
CASCADE
);

CREATE TABLE animal_photo
(
    id integer NOT NULL PRIMARY KEY REFERENCES "image" (id) ON UPDATE CASCADE ,
    id_auction integer NOT NULL REFERENCES auction (id) ON UPDATE CASCADE ON
DELETE CASCADE
);

-----------------------------------------
-- INDEXES
-----------------------------------------

 CREATE INDEX search_auction ON auction USING GIST (to_tsvector('english', name ||
' ' || species_name || ' ' || description ));

 CREATE INDEX admin_search ON "user" USING GIST (to_tsvector('english', name || '
' || email));

 CREATE INDEX notification_id ON "notification" USING hash(id_buyer);

CREATE INDEX watchlists_id ON watchlists USING hash(id_buyer);

CREATE INDEX auction_id ON auction USING btree(id_seller);

CREATE INDEX bids_id ON bids USING hash(id_buyer);

-----------------------------------------
```

```
-- TRIGGERS
-----------------------------------------

CREATE FUNCTION notify_winner() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF(NEW.id_status <> OLD.id_status)
    THEN
        INSERT INTO "notification" ("message", "read", id_auction, id_buyer)
        SELECT 'You won an auction!', FALSE, info.id_auction, info.id_buyer
        FROM (SELECT value, id_auction, id_buyer
                FROM  bids
                WHERE id_auction = NEW.id
                ORDER BY value DESC
                LIMIT 1
            ) AS info;
    END IF;
    RETURN NULL;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER notify_winner
    AFTER UPDATE ON auction
    FOR EACH ROW
    EXECUTE PROCEDURE notify_winner();


CREATE FUNCTION update_current_price() RETURNS TRIGGER AS
$BODY$
BEGIN
    UPDATE auction
        SET current_price = NEW.value
        WHERE id = NEW.id_auction;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER update_current_price
    AFTER INSERT ON bids
    FOR EACH ROW
    EXECUTE PROCEDURE update_current_price();

CREATE FUNCTION verify_bid_value() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF EXISTS (SELECT * FROM auction WHERE NEW.id_auction = id AND current_price >
NEW.value )
        THEN RAISE EXCEPTION 'A bid can only be made with a value greater than the
current bid';
    END IF;
    RETURN NEW;
END
```

```
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER verify_bid_value
    BEFORE INSERT ON bids
    FOR EACH ROW
    EXECUTE PROCEDURE verify_bid_value();


CREATE FUNCTION update_rating() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF (NEW.rating_seller >= 1 AND NEW.rating_seller <= 5)
    THEN
        UPDATE seller
        SET rating = (
            SELECT AVG(rating_seller)
            FROM auction
            WHERE auction.id = NEW.id AND auction.rating_seller >= 1 AND
auction.rating_seller <= 5
        )
        WHERE seller.id = NEW.id_seller;
    END IF;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER update_rating
    AFTER INSERT OR UPDATE ON auction
    FOR EACH ROW
    EXECUTE PROCEDURE update_rating();

CREATE FUNCTION send_notification() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF EXISTS(SELECT * FROM bids WHERE id_auction = NEW.id_auction)
    THEN
        INSERT INTO "notification" ("message", "read", id_auction, id_buyer)
        SELECT 'Your bid has been surpassed', FALSE, info.id_auction,
info.id_buyer
        FROM (SELECT value, id_auction, id_buyer
                FROM  bids
                WHERE id_auction = NEW.id_auction
                ORDER BY value DESC
                LIMIT 1
            ) AS info;
    END IF;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER send_notification
```

```
    BEFORE INSERT ON bids
    FOR EACH ROW
    EXECUTE PROCEDURE send_notification();

CREATE FUNCTION stop_ongoing_auctions() RETURNS TRIGGER AS
$BODY$
BEGIN
    UPDATE auction
        SET id_status = 2
        WHERE id IN (SELECT id FROM auction WHERE id_seller = NEW.id_seller) AND
id_status = 0;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER stop_ongoing_auctions
    AFTER INSERT ON blocks
    FOR EACH ROW
    EXECUTE PROCEDURE stop_ongoing_auctions();

CREATE FUNCTION create_buyer() RETURNS TRIGGER AS
$BODY$
BEGIN
    INSERT INTO buyer(id) VALUES (NEW.id);
    RETURN NULL;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER create_buyer
    AFTER INSERT ON "user"
    FOR EACH ROW
    EXECUTE PROCEDURE create_buyer();


CREATE FUNCTION create_seller() RETURNS TRIGGER AS
$BODY$
BEGIN
    IF NOT EXISTS(SELECT * FROM auction WHERE NEW.id = id)
       THEN INSERT INTO seller(id) values (NEW.id) ;
    END IF;
    RETURN NEW;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER create_seller
    BEFORE INSERT ON auction
    FOR EACH ROW
    EXECUTE PROCEDURE create_seller();


CREATE FUNCTION remove_watchlists() RETURNS TRIGGER AS
```

```
$BODY$
BEGIN
    IF EXISTS(SELECT * FROM auction WHERE id_status = NEW.id_status AND
NEW.id_status = 1)
    THEN
        DELETE FROM watchlists
        WHERE id_buyer = (SELECT id_buyer FROM watchlists WHERE
watchlists.id_auction = NEW.id);
    END IF;
    RETURN NULL;
END
$BODY$
LANGUAGE plpgsql;

CREATE TRIGGER remove_watchlists
    AFTER UPDATE ON auction
    FOR EACH ROW
    EXECUTE PROCEDURE remove_watchlists();
```

5.2. Database population

```
INSERT INTO "user" ("name",email,hashed_password) VALUES
    ('Dante Copeland','scelerisque@maurisut.com','FXU64FUN0IB'),
    ('Kevin Sandoval','lectus.Nullam.suscipit@senectuset.net','OJU69NCH6XM'),
    ('Nolan Y. Morgan','arcu@auctorodioa.net','PZN13XLL1XF'),
    ('Dean Galloway','ipsum.sodales.purus@Cras.com','JRD25APR5CM'),
    ('Anthony N. Wilcox','varius.ultrices.mauris@orciDonec.ca','SOZ93UJO8CZ'),
    ('Derek Stone','semper.cursus.Integer@feugiat.org','RHK73GWY1DV'),
    ('Octavius Wall','non.bibendum@massalobortis.edu','KPK52BSN6UK'),
    ('Paki J. Sutton','Morbi.metus.Vivamus@velitjusto.com','MJK94CQP2WW'),
    ('September I.
Wagner','porttitor.vulputate.posuere@blandit.co.uk','QDI49NAO1MU'),
    ('Nathaniel V. Berry','ipsum.primis.in@eratnequenon.org','RKO45HVO4TR'),
    ('Zahir R.
Blevins','nibh.dolor.nonummy@eratnonummyultricies.com','QNY68JNT1CX'),
    ('Kamal N. Cortez','nec@faucibus.co.uk','NVV84EYY8DF'),
    ('Daria X. Warren','justo.sit@sociis.ca','OTR96MXC1NE'),
    ('Natalie K. Petty','amet.diam.eu@indolor.ca','UJC45JTC5XY'),
    ('Armand H. Frye','interdum.ligula.eu@maurissapien.co.uk','DUD29LEI0IT'),
    ('Norman Q. Morrison','vitae@Aeneaneuismod.org','YHD15XGY3UH'),
    ('Lawrence Dominguez','non.luctus.sit@augueac.org','GDQ82BCE1KV'),
    ('Tanya Rios','lectus.justo@estMauris.co.uk','JRJ47SWH3JE'),
    ('Macaulay Haney','in.molestie.tortor@diamlorem.com','UAA83MYF7YN'),
    ('Harrison I. Mcclain','commodo.hendrerit@Donecestmauris.ca','RKD57XCG5OA'),
    ('Brenden C. Lawrence','molestie.dapibus.ligula@montes.edu','CYU23THB5HL'),
    ('Kiona Nielsen','nec.quam.Curabitur@dapibusid.edu','CEV17PXG6IY'),
    ('Anika Alston','purus@scelerisquesed.co.uk','KYX29KUX2RU'),
    ('Shelly T. Hoover','auctor@eget.com','RHF54MDD1QE'),
    ('Phillip D. Woodard','eu@elementum.com','FTP14TRR3BZ'),
```

```
    ('Hanna Mccarty','luctus.et.ultrices@euaccumsan.net','YCN15UXT6TG'),
    ('Gage D.
Powell','Aliquam.rutrum@tristiquepharetraQuisque.edu','QPV68CSX9LF'),
    ('Chester S. Willis','enim.nec.tempus@ipsumnuncid.com','VUU96HNF8EY'),
    ('Salvador O. Good','dui.quis@Phasellusornare.edu','HZW18IYP5CB'),
    ('Ainsley K. Castillo','feugiat@augueeu.co.uk','WNT68SOM0OA'),
    ('Sloane A. Richards','Nullam@volutpatnuncsit.co.uk','ZIP74AWK7IN'),
    ('Cally T. Rodriguez','ipsum@acfacilisis.ca','TBC40TDP6GJ'),
    ('Aline L. Bryant','nunc.est.mollis@egetlacus.net','YZV68ZJX8SW'),
    ('Katelyn Booth','dui@eget.ca','FVY06FFU4VU'),
    ('Wylie N. Osborn','amet.dapibus@magnaNamligula.com','QYI93BYJ2JN'),
    ('Rosalyn W. Tyson','sit@in.net','QCR48BUR3LZ'),
    ('Destiny T. Dean','Curabitur.massa.Vestibulum@ac.ca','CQO12BQJ8OS'),
    ('Brenden J. Patton','amet.faucibus.ut@enim.edu','XUG61VTA6EQ'),
    ('Edan Franks','justo@interdumligulaeu.co.uk','CKC15IVI8AQ'),
    ('Jasmine Cummings','facilisis@Sedcongue.org','RNS54ARC2RO'),
    ('Angela Austin','nec@musProinvel.com','UTW02MDS3QZ'),
    ('Dale E. Briggs','aliquet.sem@elitpellentesque.com','FBF97WMJ2JL'),
    ('Ishmael G. Reid','molestie.tortor.nibh@Sed.edu','RSB44PRF4QO'),
    ('Georgia G. King','massa.lobortis.ultrices@lectuspedeet.net','IBN55PXL3WJ'),
    ('Jack W. Poole','magna.Sed.eu@leoin.ca','KBK26TPU5AY'),
    ('Christen Cochran','varius@suscipitest.co.uk','LAL72OBH9SE'),
    ('Moses Hart','Donec@dis.edu','IVA23UZD1JT'),
    ('Rooney P. Harrington','nisl.Maecenas@ornare.ca','QBC48HRJ4IG'),
    ('Chaim O. Parrish','mus@lectusCumsociis.net','ILK34IXW0MZ'),
    ('Lacy X. Wiley','Duis.dignissim@ipsumcursusvestibulum.edu','SJZ34DSG7UY');

INSERT INTO "admin" (id) VALUES (1),(2),(3),(4),(5),(6),(7),(8),(9),(10);

INSERT INTO skill (TYPE) VALUES
    ('Climbs'),
    ('Jumps'),
    ('Talks'),
    ('Skates'),
    ('Olfaction'),
    ('Moonlight Navigation'),
    ('Echolocation'),
    ('Acrobatics');

INSERT INTO main_color (TYPE) VALUES
    ('Blue'),
    ('Brown'),
    ('Black'),
    ('Yellow'),
    ('Green'),
    ('Red'),
    ('White');

INSERT INTO development_stage (TYPE) VALUES
    ('Baby'),
    ('Child'),
    ('Teen'),
    ('Adult'),
    ('Elderly');
```

```
INSERT INTO category (TYPE) VALUES
    ('Mammals'),
    ('Insects'),
    ('Reptiles'),
    ('Fishes'),
    ('Birds'),
    ('Amphibians');

INSERT INTO payment_method (TYPE) VALUES
    ('Debit Card'),
    ('PayPal');

INSERT INTO shipping_method (TYPE) VALUES
    ('Standard Mail'),
    ('Express Mail'),
    ('Urgent Mail');

INSERT INTO auction_status (id,TYPE) VALUES
    (0,'Ongoing'),
    (1,'Finished'),
    (2,'Cancelled');

INSERT INTO auction
("name","description",species_name,age,starting_price,buyout_price,current_price,e
nding_date,rating_seller,id_category,id_main_color,id_dev_stage,id_payment_method,
id_shipping_method,id_seller,id_winner, id_status) VALUES
    ('Brett','vitae nibh. Donec est mauris, rhoncus id, mollis nec, cursus a,
enim. Suspendisse aliquet, sem','Jorden',6,599,13892,2998,'2020-04-
30',NULL,1,4,1,1,2,42,NULL, 0),
    ('Michael','odio. Phasellus at augue id ante dictum cursus. Nunc mauris elit,
dictum eu, eleifend nec, malesuada ut, sem. Nulla','Kylan',14,103,7313,2257,'2020-
11-30',NULL,6,2,4,1,1,43,NULL, 0),
    ('Cleo','non, egestas a, dui. Cras pellentesque. Sed dictum. Proin eget odio.
Aliquam vulputate','Sylvester',11,704,8681,6657,'2020-05-
05',NULL,2,7,2,1,3,42,NULL, 0),
    ('Ross','tristique senectus et netus et malesuada fames ac turpis egestas.
Fusce aliquet magna a neque.','Linus',6,111,9113,4449,'2020-05-
19',NULL,6,5,2,2,3,35,NULL, 0),
    ('Clementine','sem. Pellentesque ut ipsum ac mi eleifend egestas. Sed
pharetra, felis eget varius ultrices, mauris ipsum porta elit, a
feugiat','Sydnee',11,71,7457,1475,'2020-05-02',NULL,2,1,5,1,2,41,NULL,0),
    ('Elliott','porttitor eros nec tellus. Nunc lectus pede, ultrices a, auctor
non, feugiat nec, diam. Duis mi enim, condimentum','Joy',1,898,11728,1376,'2020-
06-30',NULL,2,3,3,2,2,32,NULL,0),
    ('Buffy','montes, nascetur ridiculus mus. Proin vel arcu eu odio tristique
pharetra. Quisque ac','Ashton',3,892,11497,5372,'2020-05-05',4,4,5,4,2,3,34,20,1),
    ('Kennedy','metus urna convallis erat, eget tincidunt dui augue eu tellus.
Phasellus elit pede, malesuada','Logan',15,549,14170,3679,'2020-07-
21',NULL,1,3,4,2,1,34,NULL,0),
    ('Leah','felis. Nulla tempor augue ac ipsum. Phasellus vitae mauris
sit','Freya',1,946,14533,3089,'2020-07-03',NULL,6,6,3,1,1,44,NULL,0),
    ('Cruz','nibh. Phasellus nulla. Integer vulputate, risus a ultricies
adipiscing, enim mi','Alec',1,293,13098,2780,'2020-06-
```

```
24',NULL,6,2,1,2,1,49,NULL,0),
    ('Gay','sit amet nulla. Donec non justo. Proin non massa non ante bibendum
ullamcorper. Duis cursus, diam','Laith',8,963,7624,5545,'2020-09-
12',NULL,1,5,2,2,1,31,NULL,0),
    ('Keith','Fusce mi lorem, vehicula et, rutrum eu, ultrices sit
amet,','Maxwell',7,508,13764,3711,'2020-04-29',NULL,5,1,4,2,3,35,NULL,0),
    ('Rhoda','tincidunt aliquam arcu. Aliquam ultrices iaculis odio. Nam interdum
enim non nisi. Aenean eget','Otto',14,591,9651,2187,'2020-03-
25',5,6,1,2,2,2,42,28,1),
    ('Troy','ridiculus mus. Aenean eget magna. Suspendisse tristique neque
venenatis lacus. Etiam bibendum fermentum metus. Aenean sed pede nec
ante','Nichole',10,956,9538,4886,'2020-04-23',NULL,1,4,5,2,2,48,NULL,0),
    ('Thaddeus','quis urna. Nunc quis arcu vel quam dignissim pharetra. Nam ac
nulla. In tincidunt congue turpis. In condimentum.','Steel',5,232,8737,1528,'2020-
05-06',NULL,2,1,3,1,1,34,NULL,0),
    ('Lenore','ultrices posuere cubilia Curae; Donec tincidunt. Donec vitae erat
vel pede blandit congue. In scelerisque scelerisque dui. Suspendisse ac
metus','Yolanda',2,229,11475,4017,'2020-02-12',4,5,5,5,2,3,41,27,1),
    ('Faith','suscipit, est ac facilisis facilisis, magna tellus faucibus leo, in
lobortis tellus justo sit amet nulla. Donec non justo.
Proin','Gary',3,107,11486,2201,'2020-05-08',NULL,4,6,2,2,3,43,NULL,0),
    ('Sean','aliquet magna a neque. Nullam ut nisi a odio semper cursus.
Integer','Quynn',5,138,10108,1442,'2020-02-29',2,3,5,3,2,1,48,27,1),
    ('Martin','sed pede nec ante blandit viverra. Donec tempus, lorem fringilla
ornare placerat, orci lacus vestibulum lorem, sit amet
ultricies','Kimberly',14,309,14801,4709,'2020-08-20',NULL,2,4,2,2,1,33,NULL,0),
    ('Carl','sagittis. Nullam vitae diam. Proin dolor. Nulla semper tellus id nunc
interdum feugiat.','Courtney',3,728,13671,5554,'2020-01-19',4,3,2,3,1,3,46,12,1),
    ('Jelani','non, hendrerit id, ante. Nunc mauris sapien, cursus in, hendrerit
consectetuer, cursus et, magna. Praesent
interdum','Gannon',7,494,14401,5747,'2020-01-13',4,3,6,5,2,1,33,26,1),
    ('Garth','pretium et, rutrum non, hendrerit id, ante. Nunc mauris sapien,
cursus in, hendrerit consectetuer, cursus et,','Kyle',3,513,13391,3031,'2020-10-
06',NULL,6,1,4,1,3,46,NULL,0),
    ('Raja','ullamcorper, velit in aliquet lobortis, nisi nibh lacinia orci,
consectetuer euismod est arcu ac orci. Ut semper pretium neque.
Morbi','Lani',6,565,12840,1515,'2020-04-28',NULL,3,4,1,2,2,50,NULL,0),
    ('Emmanuel','sem egestas blandit. Nam nulla magna, malesuada vel, convallis
in, cursus et, eros. Proin ultrices. Duis','Kato',8,151,8110,1761,'2020-05-
19',NULL,2,3,4,2,3,37,NULL,0),
    ('Demetria','magna nec quam. Curabitur vel lectus. Cum sociis natoque
penatibus et magnis dis','Erich',4,122,11762,6346,'2020-05-
15',NULL,2,4,5,1,2,41,NULL,0),
    ('Alexandra','Donec fringilla. Donec feugiat metus sit amet ante. Vivamus non
lorem vitae odio sagittis semper. Nam tempor diam
dictum','Hanna',4,387,8698,6773,'2020-06-06',NULL,5,7,2,2,2,49,NULL,0),
    ('Faith','Nam ac nulla. In tincidunt congue turpis. In condimentum. Donec at
arcu. Vestibulum ante ipsum','Henry',14,56,8878,3568,'2020-02-
14',2,5,3,4,1,2,39,12,1),
    ('Willa','Donec tincidunt. Donec vitae erat vel pede blandit congue.
In','Elmo',5,621,7317,2290,'2020-04-24',NULL,2,7,1,1,3,44,NULL,0),
    ('Stewart','amet, consectetuer adipiscing elit. Aliquam auctor, velit eget
laoreet posuere, enim nisl','Geoffrey',4,592,7582,5397,'2020-06-
01',NULL,6,2,4,1,2,47,NULL,0),
```

```
    ('Deacon','feugiat. Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
Aliquam auctor, velit eget laoreet posuere, enim nisl
elementum','Clio',12,135,12396,5688,'2020-02-21',4,5,2,5,2,3,50,11,1),
    ('Melissa','commodo at, libero. Morbi accumsan laoreet ipsum. Curabitur
consequat, lectus sit','Phillip',5,206,14538,6959,'2020-06-
08',NULL,3,7,3,2,3,40,NULL,0),
    ('Patrick','ridiculus mus. Donec dignissim magna a tortor. Nunc commodo auctor
velit. Aliquam nisl. Nulla eu neque pellentesque
massa','Arsenio',2,697,8201,5348,'2020-05-12',NULL,6,6,3,2,2,49,NULL,0),
    ('Jakeem','felis, adipiscing fringilla, porttitor vulputate, posuere
vulputate, lacus. Cras interdum. Nunc sollicitudin commodo ipsum. Suspendisse non
leo.','Keelie',2,890,7622,6616,'2020-02-13',5,6,1,2,1,2,48,22,1),
    ('Gareth','consectetuer ipsum nunc id enim. Curabitur massa. Vestibulum
accumsan neque et nunc. Quisque ornare tortor at risus.
Nunc','Chandler',14,675,13898,3348,'2020-03-23',4,3,1,3,2,2,42,19,1),
    ('Baxter','Quisque tincidunt pede ac urna. Ut tincidunt vehicula risus. Nulla
eget metus eu erat','Anne',4,877,10301,3249,'2020-04-
29',NULL,6,4,1,1,3,44,NULL,0),
    ('Amelia','Curabitur consequat, lectus sit amet luctus vulputate, nisi sem
semper erat, in consectetuer ipsum nunc id enim. Curabitur
massa.','Gavin',9,317,14259,6220,'2020-04-28',NULL,2,2,4,1,3,38,NULL,0),
    ('Beck','ultrices. Vivamus rhoncus. Donec est. Nunc ullamcorper, velit in
aliquet lobortis, nisi nibh lacinia orci, consectetuer euismod
est','Renee',11,289,7274,1080,'2020-06-05',NULL,6,3,2,2,1,40,NULL,0),
    ('Hedwig','velit. Aliquam nisl. Nulla eu neque pellentesque massa lobortis
ultrices. Vivamus rhoncus. Donec est. Nunc','Emi',11,889,14626,6131,'2020-04-
21',NULL,3,4,2,2,3,38,NULL,0),
    ('Hadley','orci quis lectus. Nullam suscipit, est ac facilisis facilisis,
magna','Cassidy',5,39,10736,3187,'2020-07-29',NULL,3,7,2,1,2,40,NULL,0),
    ('Kevin','nisl. Maecenas malesuada fringilla est. Mauris eu turpis. Nulla
aliquet.','Ariana',7,944,9565,2329,'2020-05-02',NULL,6,1,3,2,1,41,NULL,0),
    ('Larissa','egestas. Fusce aliquet magna a neque. Nullam ut nisi a
odio','Elvis',9,411,8574,5671,'2020-11-11',NULL,3,4,2,1,3,38,NULL,0),
    ('Dolan','at, velit. Cras lorem lorem, luctus ut, pellentesque eget, dictum
placerat, augue. Sed','Clayton',6,193,9809,6147,'2020-02-24',3,3,6,5,2,3,40,21,1),
    ('Jackson','Aenean gravida nunc sed pede. Cum sociis natoque penatibus et
magnis dis','Nevada',12,552,9422,6180,'2020-02-20',1,2,4,5,1,3,37,15,1),
    ('Guy','pede. Suspendisse dui. Fusce diam nunc, ullamcorper eu, euismod ac,
fermentum vel, mauris. Integer sem elit,
pharetra','Jordan',14,722,9702,3991,'2020-01-17',3,1,4,1,1,1,33,21,1),
    ('Adam','In mi pede, nonummy ut, molestie in, tempus eu, ligula. Aenean
euismod mauris eu elit. Nulla','Paul',6,52,13816,3005,'2020-09-
14',NULL,2,3,5,1,2,35,NULL,0),
    ('Hanna','gravida mauris ut mi. Duis risus odio, auctor vitae, aliquet nec,
imperdiet','Hector',3,97,11498,1673,'2020-04-23',NULL,4,7,5,2,2,40,NULL,0),
    ('Aphrodite','montes, nascetur ridiculus mus. Proin vel arcu eu odio tristique
pharetra. Quisque ac libero nec ligula','Dominic',7,362,8974,1452,'2020-01-
10',5,1,4,1,1,3,35,11,1),
    ('Mara','cursus a, enim. Suspendisse aliquet, sem ut cursus luctus, ipsum leo
elementum sem, vitae aliquam eros','Pandora',11,721,8897,4852,'2020-01-
04',4,4,6,4,2,2,43,16,1),
    ('Erica','magnis dis parturient montes, nascetur ridiculus mus. Proin vel arcu
eu odio tristique pharetra. Quisque ac libero nec
ligula','Dorothy',2,32,9578,2737,'2020-10-21',NULL,4,5,4,1,2,40,NULL,0),
```

```
    ('Ginger','felis. Nulla tempor augue ac ipsum. Phasellus vitae mauris sit amet
lorem semper auctor. Mauris vel turpis. Aliquam
adipiscing','Cruz',9,93,14455,2111,'2020-01-05',1,1,3,5,1,3,34,29,1);

INSERT INTO bids (value,maximum,id_auction,id_buyer) VALUES
    (7311,14859,25,16),
    (10766,13942,39,21),
    (12730,14247,8,30),
    (10555,13511,42,30),
    (10503,13858,45,26),
    (10552,14557,18,23),
    (8814,13451,5,29),
    (12749,13956,26,20),
    (12413,14310,23,27),
    (9247,14547,37,26),
    (8702,13413,30,17),
    (12322,14255,30,27),
    (11984,13740,26,27),
    (12025,14983,48,29),
    (12075,14760,27,26),
    (9292,13591,28,30),
    (12660,13005,20,17),
    (10619,14074,47,15),
    (11581,14778,43,21),
    (12977,14986,21,13),
    (12133,13770,47,18),
    (11203,13170,29,17),
    (7109,13191,45,30),
    (11636,14923,40,12),
    (11113,13145,48,12),
    (9788,14332,29,28),
    (7200,13782,31,13),
    (10828,13444,39,14),
    (11731,14747,36,17),
    (9553,13193,43,27),
    (11295,13044,10,12),
    (10323,14433,3,28),
    (9417,14064,38,27),
    (12163,13453,48,19),
    (10902,13809,46,22),
    (8826,14883,1,25),
    (12788,14289,47,29),
    (11306,14867,11,27),
    (12034,14483,14,15),
    (9477,13911,23,28),
    (11462,14566,2,27),
    (7167,14730,4,20),
    (8681,14283,20,30),
    (8653,13159,5,11),
    (12412,14460,46,20),
    (12824,14990,13,12),
    (12030,14777,15,21),
    (12341,14872,16,24),
    (7931,14694,43,23),
```

```
    (10383,14684,23,24),
    (6000,6000,19, 27);

INSERT INTO "ships" (id_seller,id_shipping_method) VALUES
    (48,2),
    (40,1),
    (43,1),
    (47,1),
    (49,1),
    (31,2),
    (42,3),
    (44,2),
    (41,1),
    (39,1),
    (35,2),
    (33,3),
    (45,1),
    (34,3),
    (37,2),
    (38,3),
    (36,2),
    (32,2),
    (46,1),
    (50,3);

INSERT INTO accepts (id_seller,id_payment_method) VALUES
    (48,1),
    (40,2),
    (43,2),
    (47,1),
    (49,1),
    (31,1),
    (42,1),
    (44,1),
    (41,2),
    (39,1),
    (35,2),
    (33,1),
    (45,1),
    (34,1),
    (37,1),
    (38,2),
    (36,2),
    (32,2),
    (46,1),
    (50,1);

INSERT INTO report_status (id,TYPE) VALUES
    (0,'Pending'),
    (1,'Approved'),
    (2,'Denied');

INSERT INTO reports ("date",id_buyer,id_seller, id_status) VALUES
    ('2020-03-26',22,33, 0),
```

```
    ('2020-03-27',14,40, 0),
    ('2020-03-28',21,50, 0),
    ('2020-03-20',26,34, 0),
    ('2020-03-21',25,45, 0),
    ('2020-03-22',19,43, 0),
    ('2020-03-30',12,31, 0),
    ('2020-03-23',16,39, 0),
    ('2020-03-24',12,40, 0),
    ('2020-03-25',24,43, 0),
    ('2020-03-28',13,43, 0),
    ('2020-03-09',22,41, 0),
    ('2020-03-13',16,44, 0),
    ('2020-03-16',12,45, 0),
    ('2020-03-02',29,41, 0),
    ('2020-03-12',12,46, 0),
    ('2020-03-15',23,48, 0),
    ('2020-03-19',22,43, 0),
    ('2020-03-22',11,44, 0),
    ('2020-03-20',24,47, 0);

INSERT INTO watchlists (id_auction,id_buyer) VALUES
    (17,11),
    (47,12),
    (44,13),
    (33,14),
    (35,15),
    (17,16),
    (18,17),
    (44,18),
    (47,20),
    (26,23),
    (19,24),
    (46,26),
    (7,28),
    (47,29),
    (42,30);


INSERT INTO blocks (end_date,id_admin,id_seller) VALUES
    ('2020-05-27',3,40),
    ('2020-06-12',3,41),
    ('2020-05-15',3,42);

INSERT INTO features (id_auction,id_skill) VALUES
    (48,6),
    (38,6),
    (50,5),
    (6,1),
    (2,7),
    (37,2),
    (11,4),
    (38,2),
    (22,2),
    (45,8),
```

```
    (43,1),
    (23,1),
    (35,7),
    (39,2),
    (37,6),
    (44,5),
    (28,4),
    (9,5),
    (3,7),
    (25,4);

INSERT INTO "image" (url) VALUES
    ('../images/profile/udfohgoid'),
    ('../images/profile/dsahkhfdg'),
    ('../images/profile/dsahdigid'),
    ('../images/profile/duashigfd'),
    ('../images/profile/khytiohtj'),
    ('../images/profile/dudifodsh'),
    ('../images/profile/vjdvhefhw'),
    ('../images/profile/fdshiufds'),
    ('../images/profile/qwewqrrtt'),
    ('../images/profile/rtyuidfgh'),
    ('../images/profile/qzxaxrete'),
    ('../images/profile/tjeriyhtr'),
    ('../images/profile/ghipigptr'),
    ('../images/profile/bfgbfbgjf'),
    ('../images/profile/iyp5ou565'),
    ('../images/profile/rwhyri2yi'),
    ('../images/profile/kl7iuojub'),
    ('../images/profile/rewhrieww'),
    ('../images/profile/fherfehif'),
    ('../images/profile/vnkdjfviu'),
    (' ../images/auction/vcxnvcxg'),
    (' ../images/auction/ruewturw'),
    (' ../images/auction/ergjroik'),
    (' ../images/auction/dasgdgas'),
    (' ../images/auction/tyuiouuy'),
    (' ../images/auction/poiuytte'),
    (' ../images/auction/çlkjhgfd'),
    (' ../images/auction/mnbvcxcv'),
    (' ../images/auction/sdfghjkj'),
    (' ../images/auction/asdfghjk'),
    (' ../images/auction/zxcvbnmv'),
    (' ../images/auction/retyuiov'),
    (' ../images/auction/xcvbnmol'),
    (' ../images/auction/qweryuio'),
    (' ../images/auction/poiyrewq'),
    (' ../images/auction/jhgfdssd'),
    (' ../images/auction/zxcvbnwu'),
    (' ../images/auction/reudtrhc'),
    (' ../images/auction/oewuiads'),
    (' ../images/auction/ujuwreoi');

INSERT INTO profile_photo (id,id_user) VALUES
```

```
    (1,6),
    (2,4),
    (3,49),
    (4,13),
    (5,22),
    (6,45),
    (7,29),
    (8,46),
    (9,5),
    (10,35),
    (11,47),
    (12,16),
    (13,11),
    (14,15),
    (15,1),
    (16,49),
    (17,19),
    (18,14),
    (19,27),
    (20,17);

INSERT INTO "animal_photo" (id,id_auction) VALUES
    (1,4),
    (2,38),
    (3,9),
    (4,45),
    (5,46),
    (6,35),
    (7,24),
    (8,49),
    (9,39),
    (10,47),
    (11,34),
    (12,6),
    (13,41),
    (14,17),
    (15,3),
    (16,25),
    (17,27),
    (18,48),
    (19,31),
    (20,30);


UPDATE auction SET rating_seller = 5 WHERE id = 50;
UPDATE auction SET id_status=1 WHERE id = 46;
```

Revision history

1. Changed Index 3 to B-Tree. Updated the SQL code, selects, updates, triggers and transactions according to the changes already made in the previous artifact.

GROUP2053, 12/04/2020

- Carlos Miguel Sousa Vieira, up201606868@fe.up.pt
- João Alberto Preto Rodrigues Praça, up201704748@fe.up.pt
- Lucas Tomás Martins Ribeiro, up201705227@fe.up.pt
- Silvia Jorge Moreira da Rocha, up201704684@fe.up.pt (Editor)