На что проверяем	Проверка	Скрипт на исправление	Пояснение
Дубли	select brand_id, brand, count(*) from brand group by brand_id, brand having count (*) > 1;	DELETE FROM brand b1 WHERE EXISTS (SELECT 1 FROM brand b2 WHERE b1.brand_id = b2.brand_id AND b1.brand = b2.brand AND b1.ctid < b2.ctid);	В этом запросе мы сравниваем каждую запись `b1` с каждой другой записью `b2` в таблице `brand` по значениям `brand_id` и `brand`. Если найдены дубликаты, то оставляем только одну запись с наименьшим значением `ctid`.
Соответствие типов	select brand_id, brand from brand where not brand_id is null and not brand_id ~ '^[0-9]+$';	UPDATE brand SET brand_id = brand, brand = brand_id WHERE not brand_id is null and not brand_id ~ '^[0-9]\u002B$';	Этот запрос обновит значения столбца `brand_id`, присвоив им значения из столбца `brand`, и наоборот, обновит значения столбца `brand`, присвоив им значения из столбца `brand_id`, для всех строк, где `brand_id` не является NULL и не соответствует шаблону '^[0-9]+$'.
Дубли	select category_id, category_name, count(*) from category group by category_id, category_name having count (*) > 1;	DELETE FROM category c1 WHERE EXISTS (SELECT 1 FROM category c2 WHERE c1.category_id = c2.category_id AND c1.category_name = c2.category_name AND c1.ctid < c2.ctid);	В этом запросе мы сравниваем каждую запись `c1` с каждой другой записью `c2` в таблице `category` по значениям `category_id` и `category_name`. Если найдены дубликаты, то оставляем только одну запись с наименьшим значением `ctid`.
Дубли	select product_id, name_short, category_id, pricing_line_id, brand_id, count(*) from product group by product_id, name_short, category_id, pricing_line_id, brand_id having count (*) > 1;	DELETE FROM product p1 WHERE EXISTS (SELECT 1 FROM product p2 WHERE p1.product_id = p2.product_id AND p1.name_short = p2.name_short AND p1.category_id = p2.category_id AND p1.pricing_line = p2.pricing_line AND p1.brand_id = p2.brand_id AND c1.ctid < c2.ctid);	В этом запросе мы сравниваем каждую запись `p1` с каждой другой записью `p2` в таблице `product` по всем значения таблицы. Если найдены дубликаты, то оставляем только одну запись с наименьшим значением `ctid`.

Ссылочная целостность	select category.category_id, product.product_id, product.category_id from category right join product on category.category_id = product.category_id where category.category_id is null ;	NSERT INTO category (category_id, category_name) SELECT product.category_id, 'не определено' FROM category RIGHT JOIN product ON category.category_id = product.category_id WHERE category.category_id IS NULL;	Этот запрос выбирает значения `product.category_id` из проверочного запроса и вставляет их в столбец `category_id` новых строк в таблице `category`. Значение "не определено" вставляется в столбец `category_name`. Запрос фильтрует только те строки, где `category.category_id` равно NULL.
Пропуски	select product_id, cost_per_item from stock where cost_per_item = '';	UPDATE stock SET cost_per_item = NULL WHERE cost_per_item = '';	Этот запрос обновляет все строки таблицы `stock`, где значение столбца `cost_per_item` равно пустой строке, и устанавливает значение этого столбца в NULL. Запрос использует условие `WHERE cost_per_item = ''`, чтобы выбрать только строки с пустым значением в столбце `cost_per_item`.
Пропуски	select product_id, cost_per_item from stock where product_id = '';	UPDATE stock SET product_id = NULL WHERE product_id = '';	Этот запрос обновляет все строки таблицы `stock`, где значение столбца `product_id` равно пустой строке, и устанавливает значение этого столбца в NULL. Запрос использует условие `WHERE product_id= ''`, чтобы выбрать только строки с пустым значением в столбце `product_id`.
Дубли	select available_on, product_id , pos, available_quantity , cost_per_item , count(*) from stock group by available_on, product_id , pos, available_quantity , cost_per_item having count (*) > 1;	DELETE FROM stock s1 WHERE EXISTS (SELECT 1 FROM stock s2 WHERE s1.available_on = s2.available_on AND s1.product_id= s2.product_id AND s1.pos = s2.pos AND s1.available_quantity = s2.available_quantity AND s1.cost_per_item = s2.cost_per_item AND c1.ctid < c2.ctid);	В этом запросе мы сравниваем каждую запись `s1` с каждой другой записью `s2` в таблице `stock` по всем значения таблицы. Если найдены дубликаты, то оставляем только одну запись с наименьшим значением `ctid`.

Ссылочная целостность	select product.product_id, stock.product_id from stock left join product on stock.product_id = product. product_id where product.product_id is null ;	INSERT INTO product (product_id, name_short) SELECT stock.product_id, 'не определено' FROM stock LEFTJOIN product ON stock.product_id = product.product_id WHERE product.product_id IS NULL;	Этот запрос выбирает значения `stock. product_id` из проверочного запроса и вставляет их в столбец `product_id` новых строк в таблице `product`. Значение "не определено" вставляется в столбец `name_short`. Запрос фильтрует только те строки, где `product.product_id` равно NULL.
выбросы	select MIN(cast(available_quantity as FLOAT)) as min_quantity, MAX(cast(available_quantity as FLOAT)) as max_quantity, PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY cast(available_quantity as FLOAT)) as median_quantity from stock where available_quantity <> '';	UPDATE stock SET available_quantity = NULL WHERE available_quantity < 0;	Этот запрос обновляет значения столбца `available_quantity` на `NULL` для всех строк в таблице `stock`, где значение столбца `available_quantity` меньше 0. Запрос использует условие `WHERE available_quantity < 0`, чтобы выбрать только строки с отрицательным значением в столбце `available_quantity`.
Пропуски	select * from sources."transaction" where transaction_id = '' or product_id = '' or recorded_on = '' or quantity = '' or price = '' or price_full = '' or order_type_id = '';	UPDATE transactions SET transaction_id= NULL WHERE transaction_id= ''; UPDATE transactions SET product_id = NULL WHERE product_id = ''; UPDATE transactions SET recorded_on= NULL WHERE recorded_on = ''; UPDATE transactions SET quantity = NULL WHERE quantity= ''; UPDATE transactions SET order_type_id = NULL WHERE order_type_id = '';	Эти запросы обновляют все строки таблицы `transaction`, где значение столбцов равно пустой строке, и устанавливает значение этого столбца в NULL. Запрос использует условие `WHERE product_id= '' , чтобы выбрать только строки с пустым значением в столбце `product_id`, и аналогично с любым другим столбцом.

Ссылочная целостность	select sources.product.product_id, sources."transaction".product_id from sources.transaction left join sources.product on sources."transaction".product_id = product.product_id where product.product_id is null ;	INSERT INTO product (product_id, name_short) SELECT transaction.product_id, 'не определено' FROM transaction LEFTJOIN product ON transaction.product_id = product.product_id WHERE product.product_id IS NULL;	Этот запрос выбирает значения `transaction.product_id` из проверочного запроса и вставляет их в столбец `product_id` новых строк в таблице `product`. Значение "не определено" вставляется в столбец `name_short`. Запрос фильтрует только те строки, где `product.product_id` равно NULL.