Государственный Университет Молдовы   
Факультет Математики и Информатики  
Департамент Информатики

**Индивидуальная работа**

По курсу «Базы данных»

тема: Создание базы данных «Logistics\_Hub\_DB»

Выполнил студент группа I2302:

Богданов Юрий  
Проверила преподаватель:  
Карчева Н.Ф.

Кишинёв, 2024

**Цель работы:**

Создание реляционной базы данных «Logistics\_Hub\_DB» в приложении Oracle APEX, предназначенной для управления данными о дальнобойщиках, маршрутах, грузах и грузовиках. Данная база данных необходима для регистрации и учета данных водителей, хранения информации о моделях грузовиков, их грузоподъемности, пробеге и состоянии, учета данных о типах грузов, их весе, стоимости, хрупкости и страховке, хранения данных о маршрутах, включая города отправления и назначения, а также расстояния между пунктами. База данных позволяет отслеживать выполнение доставок, включая данные о водителях, грузовиках, маршрутах и статусах, что способствует мониторингу выполнения задач в режиме реального времени. Также база данных предоставляет возможности для выполнения сложных аналитических запросов, таких как подсчёт доходов, затрат на обслуживание и топливо, определение популярных маршрутов и долгосрочных тенденций в перевозках. База данных предназначена для оптимизации логистических процессов, снижения эксплуатационных затрат и повышения эффективности управления транспортной компанией.

1) Отношения находятся в нормальной форме.

2) Ограничения структуры (уникальность ключа).

3) Реляционные операторы.

**Этапы создания базы данных:**

1. **Анализ предметной области.**

В ходе анализа были выделены ключевые элементы логистической системы дальнобойщиков, включая учет водителей, управление грузовиками, маршруты, грузы, доставки, обслуживание транспортных средств и затраты на топливо. Каждая из этих сущностей требует отдельной таблицы для организации данных и их обработки.

1. **Сбор необходимой информации.**

Собрана информация о необходимых атрибутах для каждой сущности. Например, для водителей это имя, номер водительского удостоверения, опыт работы и рейтинг; для грузовиков — модель, номерной знак, грузоподъемность и пробег; для маршрутов — города отправления и назначения, а также расстояние; для грузов — описание, вес, стоимость и хрупкость.

1. **Создание проекта**.

На основе собранной информации была разработана логическая структура базы данных, включающая таблицы и связи между ними. Таблицы нормализованы для минимизации избыточности данных, а внешние ключи обеспечивают целостность данных.

1. **Физическая реализация.**

Проект был реализован в приложении Oracle APEX. Были созданы таблицы базы данных, настроены ограничения и связи, база была заполнена тестовыми данными, а также разработаны запросы и функции для работы с данными.

**Содержание работы:**

**1) Схема базы данных.**  
Схема базы данных отражает структуру всей системы, включая таблицы и связи между ними. В схеме представлены основные сущности, такие как водители, грузовики, маршруты, грузы, доставки, обслуживание и заправки. Для каждой таблицы указаны её поля, первичные и внешние ключи, а также типы данных. Схема обеспечивает целостность данных и их логическую организацию, что упрощает работу с базой.

**2) Примеры запросов для получения необходимой информации из базы данных.**  
В работе представлены примеры сложных и составных запросов, которые позволяют решать реальные задачи управления логистикой. Например, запросы для анализа загруженности водителей, популярности маршрутов, затрат на топливо и обслуживание, анализа характеристик грузов и оценки эффективности транспортировки. Запросы охватывают различные аспекты работы базы данных и демонстрируют её возможности.

**3) Создание представлений.**  
Представления используются для упрощения работы с данными и предоставления упорядоченного доступа к сложным наборам данных. В работе представлены примеры создания представлений, которые собирают информацию из нескольких таблиц, например, для отображения деталей доставок, анализа расходов на топливо, популярности маршрутов и состояния автопарка. Представления упрощают процесс получения аналитических данных и обеспечивают безопасность, скрывая детали реализации базы данных.

**Схема базы данных включает в себя следующие таблицы:**

· **Drivers** (driver\_id - ID водителя, name – имя водителя, license\_number – номер водительского удостоверения, experience – стаж работы, rating – рейтинг водителя).

· **Trucks** (truck\_id - ID грузовика, model – модель грузовика, plate\_number – номерной знак, capacity – грузоподъемность, mileage – пробег).

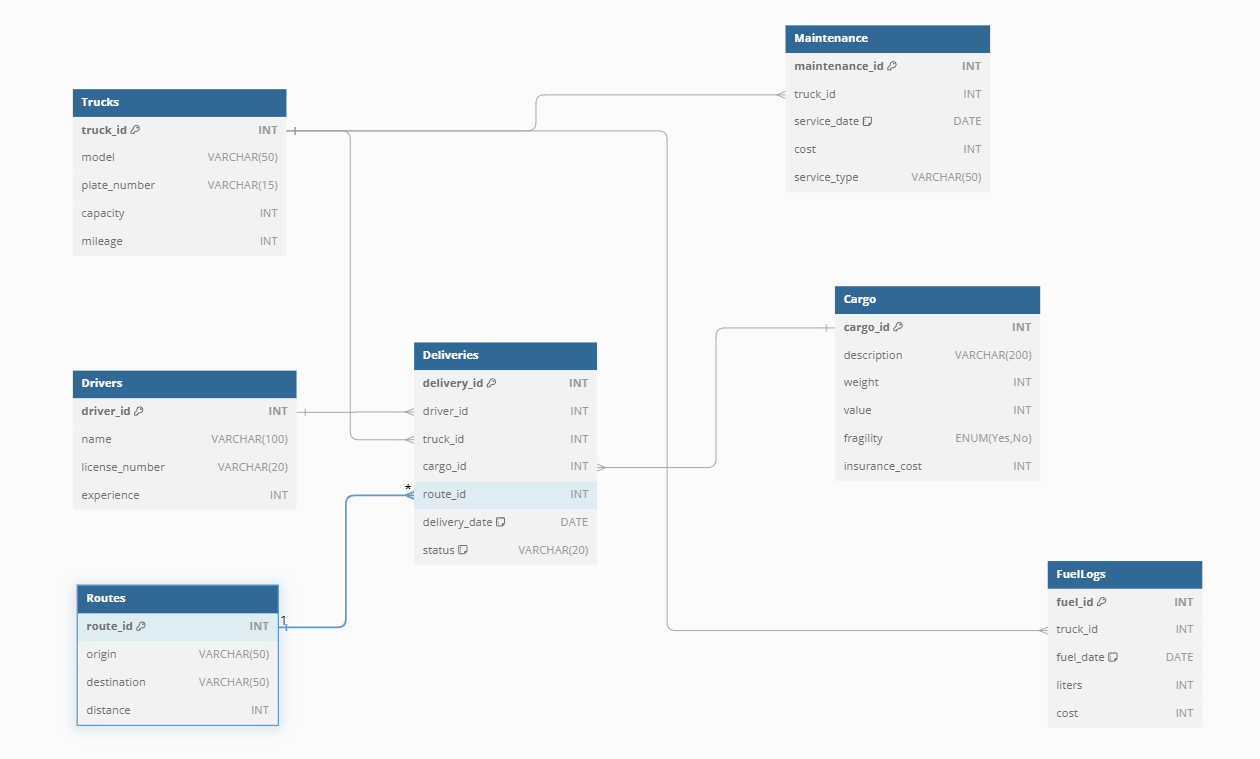
· **Cargo** (cargo\_id - ID груза, description – описание груза, weight – вес груза, value – стоимость груза, fragility – хрупкость груза, insurance\_cost – стоимость страховки).

· **Routes** (route\_id - ID маршрута, origin – город отправления, destination – город назначения, distance – длина маршрута).

· **Deliveries** (delivery\_id - ID доставки, driver\_id – ID водителя, truck\_id – ID грузовика, cargo\_id – ID груза, route\_id – ID маршрута, delivery\_date – дата доставки, status – статус доставки).

· **Maintenance** (maintenance\_id - ID записи обслуживания, truck\_id – ID грузовика, service\_date – дата обслуживания, cost – стоимость обслуживания, service\_type – тип обслуживания).

· **FuelLogs** (fuel\_id - ID записи, truck\_id – ID грузовика, fuel\_date – дата заправки, liters – количество литров, cost – стоимость топлива).



**Создание запросов(SELECT):**

1. **Вывести информацию о маршрутах, по которым доставлялись грузы, стоимость которых превышает среднюю стоимость всех грузов.**

SELECT r.route\_id, r.origin, r.destination, COUNT(del.delivery\_id) AS total\_deliveries,

SUM(c.value) AS total\_cargo\_value

FROM Routes r

JOIN Deliveries del ON r.route\_id = del.route\_id

JOIN Cargo c ON del.cargo\_id = c.cargo\_id

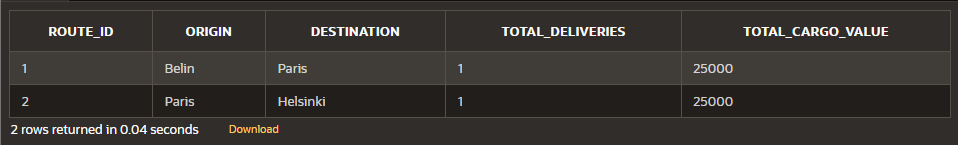
WHERE c.value > (SELECT AVG(value) FROM Cargo)

GROUP BY r.route\_id, r.origin, r.destination

HAVING SUM(c.value) > 0

ORDER BY total\_cargo\_value DESC;

Этот запрос выбирает информацию о маршрутах, по которым доставлялись грузы, стоимость которых превышает среднюю стоимость всех грузов. Таблицы Routes, Deliveries и Cargo соединяются по их ключам, чтобы связать маршруты с доставками и грузами. Фильтр WHERE ограничивает выбор только грузами с высокой стоимостью, вычисляя среднее значение стоимости всех грузов с помощью подзапроса. Затем данные группируются по маршрутам, чтобы подсчитать количество доставок и сумму стоимости грузов для каждого маршрута. Условие HAVING гарантирует, что только маршруты с положительной общей стоимостью грузов включаются в результат. Итоговый результат сортируется по убыванию общей стоимости грузов, чтобы сначала показать самые прибыльные маршруты.



**2) Найти всех водителей, которые выполнили хотя бы одну доставку с грузами весом более 2000 кг, использовали грузовики с пробегом более 100000 км, и маршруты длиной выше среднего.**

SELECT d.driver\_id, d.name, c.cargo\_id, c.description, c.weight, t.truck\_id, t.model, t.mileage, r.route\_id, r.distance

FROM Drivers d

JOIN Deliveries del ON d.driver\_id = del.driver\_id

JOIN Cargo c ON del.cargo\_id = c.cargo\_id

JOIN Trucks t ON del.truck\_id = t.truck\_id

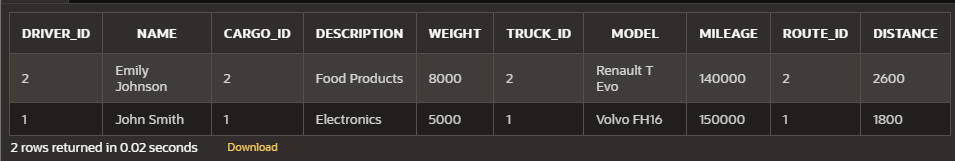
JOIN Routes r ON del.route\_id = r.route\_id

WHERE c.weight > 2000

AND t.mileage > 100000

AND r.distance > (SELECT AVG(distance) FROM Routes)

ORDER BY r.distance DESC, t.mileage DESC;  
  
Этот запрос выбирает данные о доставках, которые выполняли водители, используя грузовики с большим пробегом, и грузы с высоким весом, доставленные по маршрутам, длина которых превышает среднюю длину всех маршрутов. Таблицы Drivers, Deliveries, Cargo, Trucks и Routes соединяются, чтобы связать данные о водителях, доставках, грузах, грузовиках и маршрутах. Условие WHERE ограничивает выбор записями, где вес груза превышает 2000 кг, пробег грузовика более 100000 км, а длина маршрута больше средней длины маршрутов (среднее значение вычисляется подзапросом). Итоговый результат сортируется в порядке убывания длины маршрута, а при равных значениях длины маршрута — в порядке убывания пробега грузовиков.



**3) Запрос: Подсчитать общее количество доставок, выполненных каждым водителем, и процент доставок, в которых грузы были хрупкими ("Yes").**

SELECT d.driver\_id, d.name,

COUNT(del.delivery\_id) AS total\_deliveries,

SUM(CASE WHEN c.fragility = 'Yes' THEN 1 ELSE 0 END) AS fragile\_deliveries,

ROUND(SUM(CASE WHEN c.fragility = 'Yes' THEN 1 ELSE 0 END) \* 100.0 / COUNT(del.delivery\_id), 2) AS fragile\_percentage

FROM Drivers d

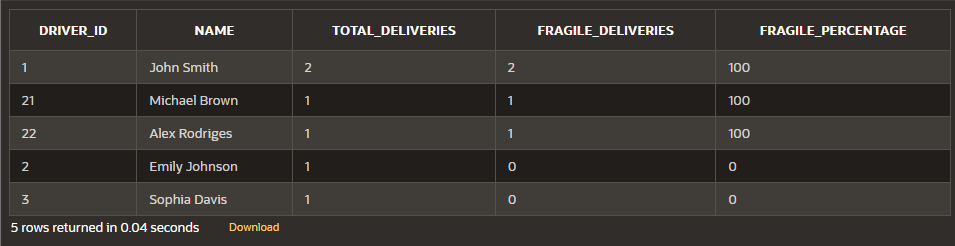
JOIN Deliveries del ON d.driver\_id = del.driver\_id

JOIN Cargo c ON del.cargo\_id = c.cargo\_id

GROUP BY d.driver\_id, d.name

HAVING COUNT(del.delivery\_id) > 0

ORDER BY fragile\_percentage DESC, total\_deliveries DESC;  
  
Этот запрос анализирует количество доставок, выполненных каждым водителем, и вычисляет процент доставок, связанных с хрупкими грузами. Таблицы Drivers, Deliveries и Cargo соединяются, чтобы связать водителей с их доставками и грузами. Для каждого водителя подсчитывается общее количество доставок с помощью агрегатной функции COUNT, а также количество доставок хрупких грузов с использованием условия CASE WHEN. Вычисляется процент доставок с хрупкими грузами относительно общего числа доставок, округлённый до двух знаков после запятой. Фильтр HAVING исключает водителей, которые не выполнили ни одной доставки. Результат сортируется сначала по проценту доставок с хрупкими грузами в порядке убывания, а затем по общему количеству доставок, также в порядке убывания.



**4) Найти топ-3 водителей по количеству доставок на маршрутах длиной более 1000 км, вывести их общий пробег грузовиков и среднюю длину маршрута.**

SELECT d.driver\_id, d.name, COUNT(del.delivery\_id) AS total\_deliveries,

SUM(t.mileage) AS total\_mileage,

AVG(r.distance) AS avg\_route\_distance

FROM Drivers d

JOIN Deliveries del ON d.driver\_id = del.driver\_id

JOIN Trucks t ON del.truck\_id = t.truck\_id

JOIN Routes r ON del.route\_id = r.route\_id

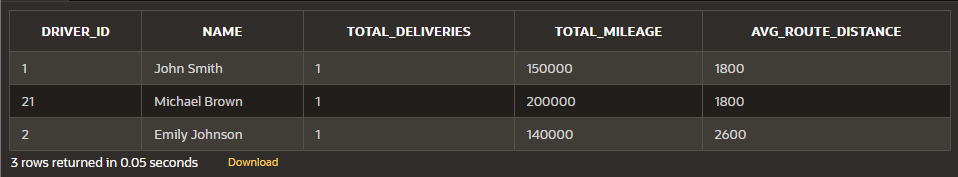
WHERE r.distance > 1000

GROUP BY d.driver\_id, d.name

ORDER BY total\_deliveries DESC

FETCH FIRST 3 ROWS ONLY;

Этот сложный запрос анализирует водителей, которые выполняли доставки на длинные маршруты (длиной более 1000 км), и подсчитывает общее количество доставок, общий пробег грузовиков, которыми они пользовались, и среднюю длину маршрутов. Таблицы Drivers, Deliveries, Trucks и Routes соединяются, чтобы связать водителей с доставками, грузовиками и маршрутами. Агрегатная функция COUNT используется для подсчёта доставок, SUM для расчёта общего пробега грузовиков, а AVG вычисляет среднюю длину маршрута. Группировка по водителям (GROUP BY) позволяет получить эти данные для каждого водителя. Условие WHERE фильтрует маршруты длиной более 1000 км. Результат сортируется в порядке убывания общего количества доставок (ORDER BY total\_deliveries DESC), и выводятся только первые три записи с помощью FETCH FIRST 3 ROWS ONLY.



**5) Найти общий доход от грузов с хрупкостью "Yes", перевезённых по маршрутам длиной более 1000 км.**

SELECT SUM(c.value) AS total\_income

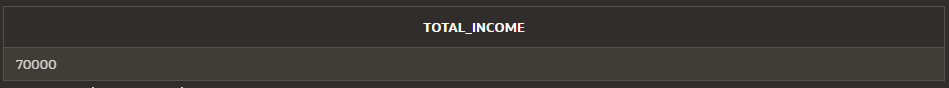
FROM Cargo c

JOIN Deliveries del ON c.cargo\_id = del.cargo\_id

JOIN Routes r ON del.route\_id = r.route\_id

WHERE c.fragility = 'Yes' AND r.distance > 1000;

Этот составной запрос вычисляет общий доход от хрупких грузов, которые доставлялись по маршрутам длиной более 1000 км. Таблицы Cargo, Deliveries и Routes соединяются, чтобы связать данные о грузах, доставках и маршрутах. Условие WHERE фильтрует записи, оставляя только хрупкие грузы (c.fragility = 'Yes') и маршруты с длиной более 1000 км (r.distance > 1000). Агрегатная функция SUM используется для расчёта общей стоимости таких грузов. Результат представляет собой одно число — суммарный доход от подходящих грузов.



**6) Найти все грузы, которые были доставлены водителями с опытом работы более 6 лет, и их маршрут.**

SELECT c.cargo\_id, c.description, d.name, r.route\_id, r.origin, r.destination

FROM Cargo c

JOIN Deliveries del ON c.cargo\_id = del.cargo\_id

JOIN Drivers d ON del.driver\_id = d.driver\_id

JOIN Routes r ON del.route\_id = r.route\_id

WHERE d.experience > 6;

Этот составной запрос выбирает информацию о грузах, которые доставлялись водителями с опытом работы более 6 лет. Таблицы Cargo, Deliveries, Drivers и Routes соединяются, чтобы связать данные о грузах, доставках, водителях и маршрутах. Условие WHERE фильтрует записи, оставляя только тех водителей, чей стаж работы (experience) превышает 6 лет. Результат содержит ID груза, описание груза, имя водителя, ID маршрута, город отправления и город назначения, что позволяет получить полное представление о таких доставках.



### **7) **Вывести грузовики, которые выполнили доставки с грузами стоимостью выше 10000.****

SELECT DISTINCT t.truck\_id, t.model, t.capacity

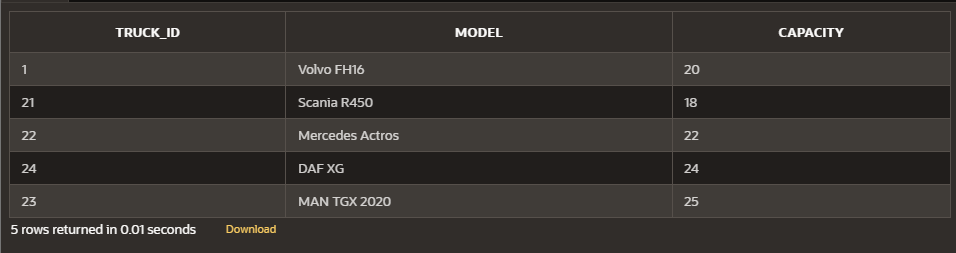
FROM Trucks t

JOIN Deliveries del ON t.truck\_id = del.truck\_id

JOIN Cargo c ON del.cargo\_id = c.cargo\_id

WHERE c.value > 10000;

Этот запрос выбирает уникальные грузовики, которые использовались для доставки грузов стоимостью более 10000. Таблицы Trucks, Deliveries и Cargo соединяются, чтобы связать данные о грузовиках, доставках и перевозимых грузах. Условие WHERE фильтрует записи, оставляя только те, где стоимость груза (c.value) превышает 10000. Оператор DISTINCT обеспечивает уникальность записей в результате, исключая дублирующиеся данные о грузовиках. Результат содержит ID грузовика, его модель и грузоподъемность.



1. **Подсчитать общее количество доставок и среднюю длину маршрутов для каждого грузовика.**

SELECT t.truck\_id, t.model, COUNT(del.delivery\_id) AS total\_deliveries, AVG(r.distance) AS avg\_distance

FROM Trucks t

JOIN Deliveries del ON t.truck\_id = del.truck\_id

JOIN Routes r ON del.route\_id = r.route\_id

GROUP BY t.truck\_id, t.model;

Этот сложный запрос анализирует данные о грузовиках и подсчитывает, сколько доставок выполнил каждый грузовик, а также вычисляет среднюю длину маршрутов, по которым выполнялись эти доставки. Таблицы Trucks, Deliveries и Routes соединяются, чтобы связать данные о грузовиках, их доставках и маршрутах. Агрегатная функция COUNT используется для подсчёта количества доставок для каждого грузовика, а функция AVG вычисляет среднюю длину маршрутов. Группировка GROUP BY по ID и модели грузовика обеспечивает расчёты для каждого уникального грузовика. Результат содержит ID грузовика, его модель, общее количество доставок и среднюю длину маршрута.



1. **Вывести маршруты, по которым длина выше 1500 км, и количество доставок для каждого водителя, который их выполнял.**

SELECT r.route\_id, r.origin, r.destination, r.distance,

d.driver\_id, d.name, COUNT(del.delivery\_id) AS delivery\_count

FROM Routes r

JOIN Deliveries del ON r.route\_id = del.route\_id

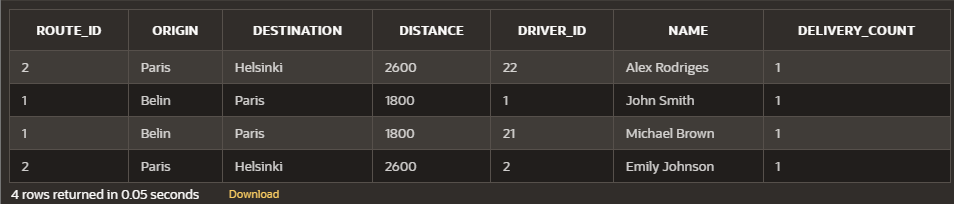
JOIN Drivers d ON del.driver\_id = d.driver\_id

WHERE r.distance > 1500

GROUP BY r.route\_id, r.origin, r.destination, r.distance, d.driver\_id, d.name

ORDER BY delivery\_count DESC;

Этот сложный запрос анализирует маршруты длиной более 1500 км и подсчитывает, сколько доставок было выполнено каждым водителем по каждому из таких маршрутов. Таблицы Routes, Deliveries и Drivers соединяются, чтобы связать маршруты с их доставками и водителями, которые их выполнили. Условие WHERE ограничивает выбор маршрутами, длина которых превышает 1500 км. Агрегатная функция COUNT подсчитывает количество доставок для каждого водителя по каждому маршруту. Группировка GROUP BY по маршруту (ID, города отправления и назначения, длина) и водителю (ID, имя) позволяет получить эти данные для каждого уникального сочетания маршрута и водителя. Результат сортируется по количеству доставок (delivery\_count) в порядке убывания, чтобы сначала показать водителей с наибольшим числом выполненных доставок.



**10) Найти все маршруты, по которым выполнялись доставки грузовиками с пробегом более 150000 км.**

SELECT DISTINCT r.route\_id, r.origin, r.destination, r.distance

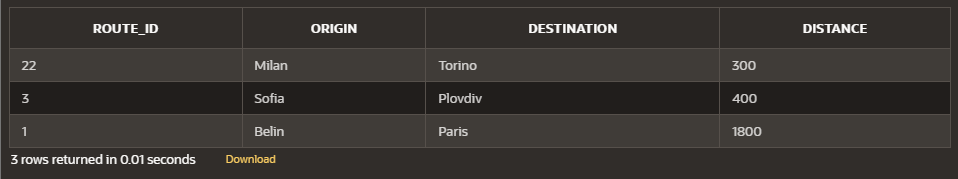
FROM Routes r

JOIN Deliveries del ON r.route\_id = del.route\_id

JOIN Trucks t ON del.truck\_id = t.truck\_id

WHERE t.mileage > 150000;

Этот составной запрос выбирает уникальные маршруты, по которым выполнялись доставки с использованием грузовиков, чей пробег превышает 150000 километров. Таблицы Routes, Deliveries и Trucks соединяются, чтобы связать данные о маршрутах, доставках и грузовиках. Условие WHERE фильтрует записи, оставляя только те, где пробег грузовика (t.mileage) превышает 150000. Оператор DISTINCT обеспечивает выбор только уникальных маршрутов, исключая повторяющиеся записи. Результат содержит ID маршрута, город отправления, город назначения и длину маршрута.



**Создание представлений(CREATE VIEW):**

1. **Информация о доставке хрупких грузов.**

CREATE OR REPLACE VIEW FragileCargoDeliveries AS

SELECT del.delivery\_id, d.name AS driver\_name, c.description AS cargo\_description, c.fragility, r.origin, r.destination, r.distance

FROM Deliveries del

JOIN Drivers d ON del.driver\_id = d.driver\_id

JOIN Cargo c ON del.cargo\_id = c.cargo\_id

JOIN Routes r ON del.route\_id = r.route\_id

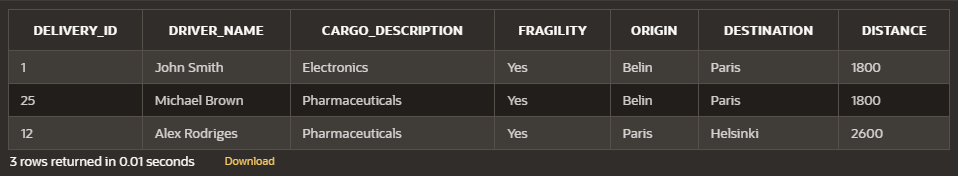
WHERE c.fragility = 'Yes';

Это представление объединяет данные о доставках, водителях, грузах и маршрутах, чтобы предоставить информацию о доставках хрупких грузов. Таблицы Deliveries, Drivers, Cargo и Routes соединяются через ключи, а условие WHERE c.fragility = 'Yes' фильтрует только те доставки, где грузы помечены как хрупкие. Результатом является виртуальная таблица, которая содержит ID доставки, имя водителя, описание груза, признак хрупкости, город отправления, город назначения и длину маршрута. Такое представление удобно для быстрого анализа данных о доставках хрупких грузов.

**Проверка:** Найти все хрупкие грузы, доставленные по маршрутам длиной более 1200 км.

SELECT \* FROM FragileCargoDeliveries

WHERE distance > 1200;



1. **Маршруты и общее количество доставок по ним.**

CREATE OR REPLACE VIEW RouteDeliveries AS

SELECT r.route\_id, r.origin, r.destination, r.distance, COUNT(del.delivery\_id) AS delivery\_count

FROM Routes r

JOIN Deliveries del ON r.route\_id = del.route\_id

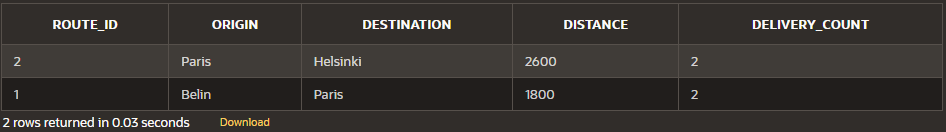
GROUP BY r.route\_id, r.origin, r.destination, r.distance;

Это представление объединяет данные о маршрутах и доставках, чтобы предоставить информацию о количестве доставок, выполненных по каждому маршруту. Таблицы Routes и Deliveries соединяются через поле route\_id, чтобы связать данные о маршрутах с их доставками. Агрегатная функция COUNT подсчитывает количество доставок для каждого маршрута. Группировка GROUP BY используется, чтобы вычисления выполнялись отдельно для каждого маршрута, основываясь на его ID, городах отправления и назначения, а также длине. Результатом является виртуальная таблица, содержащая ID маршрута, города отправления и назначения, длину маршрута и количество выполненных доставок. Это представление полезно для анализа активности маршрутов и выявления наиболее используемых.

**Проверка:** Найти маршруты, по которым выполнено более 1 доставки.

SELECT \* FROM RouteDeliveries

WHERE delivery\_count > 1;



1. **Водители и их опыт.**

CREATE OR REPLACE VIEW DriversExperience AS

SELECT d.driver\_id, d.name, d.license\_number, d.experience, COUNT(del.delivery\_id) AS total\_deliveries

FROM Drivers d

LEFT JOIN Deliveries del ON d.driver\_id = del.driver\_id

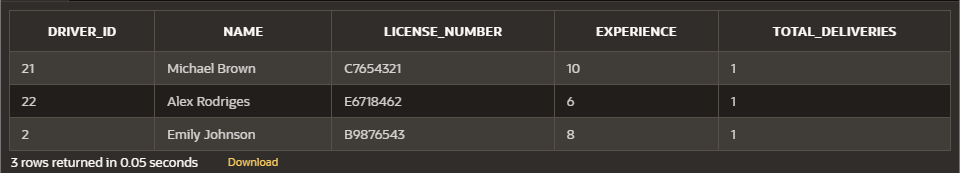
GROUP BY d.driver\_id, d.name, d.license\_number, d.experience;

Это представление объединяет данные о водителях и их доставках, чтобы предоставить информацию об общем количестве выполненных доставок и опыта для каждого водителя. Таблицы Drivers и Deliveries соединяются через поле driver\_id, используя левое соединение (LEFT JOIN), чтобы включить всех водителей, даже если они не выполнили ни одной доставки. Агрегатная функция COUNT подсчитывает количество доставок, выполненных каждым водителем. Группировка GROUP BY выполняется по ID водителя, его имени, номеру водительского удостоверения и стажу работы. Результатом является виртуальная таблица, содержащая ID водителя, его имя, номер водительского удостоверения, стаж работы и общее количество доставок. Это представление полезно для анализа активности водителей и их опыта.

**Проверка:** Найти всех водителей с опытом более 5 лет, которые выполнили хотя бы одну доставку.

SELECT \* FROM DriversExperience

WHERE experience > 5 AND total\_deliveries > 0;



1. **Заправки грузовиков.**

CREATE OR REPLACE VIEW TruckFuelLogs AS

SELECT f.truck\_id, t.model, SUM(f.liters) AS total\_liters, SUM(f.cost) AS total\_fuel\_cost

FROM FuelLogs f

JOIN Trucks t ON f.truck\_id = t.truck\_id

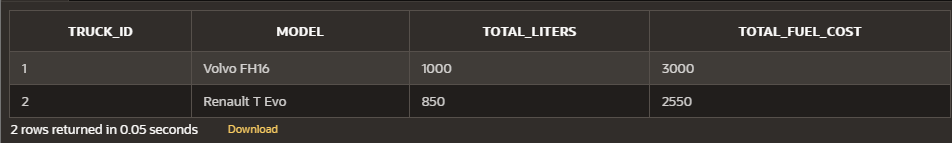
GROUP BY f.truck\_id, t.model;

Это представление собирает данные о заправках грузовиков, объединяя информацию о грузовиках и их расходе топлива. Таблицы FuelLogs и Trucks соединяются через поле truck\_id, чтобы связать данные о заправках с конкретными грузовиками. Агрегатные функции SUM используются для подсчёта общего объёма топлива (в литрах) и общей стоимости заправок для каждого грузовика. Группировка GROUP BY выполняется по ID грузовика и его модели, чтобы расчёты выполнялись отдельно для каждого уникального грузовика. Результатом является виртуальная таблица, которая включает ID грузовика, его модель, общий объём топлива и общую стоимость заправок. Это представление полезно для анализа топливных расходов грузовиков.

**Проверка:** Найти грузовики, которые заправлялись на общую сумму более 2000.

SELECT \* FROM TruckFuelLogs

WHERE total\_fuel\_cost > 2000;



1. **Грузы с высокой стоимостью.**

CREATE OR REPLACE VIEW HighValueCargo AS

SELECT c.cargo\_id, c.description, c.value, c.fragility, del.delivery\_date, r.origin, r.destination

FROM Cargo c

JOIN Deliveries del ON c.cargo\_id = del.cargo\_id

JOIN Routes r ON del.route\_id = r.route\_id

WHERE c.value > 20000;

Это представление объединяет данные о высокоценных грузах, доставках и маршрутах. Таблицы Cargo, Deliveries и Routes соединяются через их ключевые поля, чтобы связать информацию о грузах с их доставками и маршрутами. Условие WHERE c.value > 20000 фильтрует записи, оставляя только те грузы, чья стоимость превышает 20,000. Результатом является виртуальная таблица, содержащая ID груза, его описание, стоимость, признак хрупкости, дату доставки, город отправления и город назначения маршрута. Это представление удобно для анализа перевозок грузов с высокой стоимостью.

**Проверка:** Найти высокоценные грузы, доставленные после 1 ноября 2024 года.

SELECT \* FROM HighValueCargo

WHERE delivery\_date > TO\_DATE('2024-11-01', 'YYYY-MM-DD');



**Вывод**

В результате выполнения работы была создана реляционная база данных «Logistics\_Hub\_DB», которая соответствует требованиям нормализации и обеспечивает удобство работы с данными о перевозках. Разработанная структура включает 7 таблиц, описывающих основные аспекты логистической системы, такие как водители, грузовики, маршруты, грузы, доставки, обслуживание и заправки. Установлены связи между таблицами с использованием внешних ключей, что позволяет эффективно управлять данными.

В процессе работы были реализованы 10 сложных и составных SQL-запросов, которые позволяют анализировать данные о доставках, маршрутах, водителях и расходах. Запросы включают использование соединений таблиц, агрегатных функций, фильтрации и подзапросов, что делает их полезными для анализа и принятия решений. Также создано 5 представлений для упрощения доступа к агрегированным данным и оптимизации работы с базой данных. Все запросы и представления были проверены, результаты соответствуют ожиданиям.

Таким образом, поставленные задачи выполнены, и разработанная база данных может быть использована для управления и анализа процессов в сфере грузоперевозок.