

Online Courses

Introduction

As part of the project, a database has been developed that simulates the operation of an online programming course platform. There are 4 categories of courses available on the platform, such as Programming Language, Software Testing, Data Science and Analytical.

And in each of the categories you can find the corresponding courses.

There are also variations of courses after which the user receives a certificate, and there are also courses after which the certificate is not attached.

OLTP Database

What is stored in the OLTP database

The OLTP (Online Transaction Processing) database is designed to store operational and current data. In my case, the OLTP database reflects the current data of the online course platform related to users, courses, their completion and payment.

The main objects that are stored in the OLTP database:

1. Users (Users)

The `Users` table contains data about registered users of the system: a unique identifier, name, email, and registration date. This is the basic information needed to identify and interact with users.

2. Course Categories(Categories)

The `Categories` table stores course categories to group courses by topic and area. This helps to structure the course catalog and simplify navigation.

3. Course (Course)

The `Course` table contains information about each course: the name, which category it belongs to, whether it has a certificate, and the price. Courses are the main product of the platform that users take.

4. **Course records (Records)**

The `Records` table records which users started taking which courses, when it happened (`records_date`), as well as the current status of the course (for example, "in progress", "completed"). It helps to track user activity.

5. **Course Ratings (Rating)**

The `Rating` table contains user ratings for courses (from 1 to 5). This is important for analyzing the quality of courses and making recommendations.

6. **Certificates (Certificate)**

The `Certificate` table stores information about issued certificates — who received the certificate for successful completion of the course and when.

7. **Payments (Payment)**

The `Payment` table records user course payment transactions — the amount, payment date, user, and course. This is the key information for income accounting and financial statements.

8. **Instructors (Instructors) and connection with courses (Course_Instructors)**

The `Instructors` and `Course_Instructors` tables store data about instructors and which courses they teach.

9. **Temporary tables (temp_*)**

Several temporary tables are used for loading and preprocessing data (`temp_users`, `temp_categories_and_courses`, `temp_records_and_ratings`, `temp_cert_pay`, `temp_instructors`). They are used for secure and consistent data loading and transformation before insertion into the main OLTP tables.

OLAP Database

1. The purpose of the project

The goal of the project is to build an **OLAP database (Data Warehouse)** for an online course platform. The main task is – to implement an **ETL process** for transferring, processing and storing data from an **OLTP system** to an **OLAP storage** with the ability to perform analytical queries.

2. Context and analytical tasks

What is the OLAP database implemented for?

The OLAP database is designed for reliable storage and management of operational user training data on an online platform. It allows you to :

- Track and analyze the dynamics of user payments by exchange rates over time.

- Calculate the number of unique users who pay for courses each month.

- Store and calculate average grades (ratings) of courses to assess their quality.

- Determine the popularity of courses among users based on records and ratings.

- Analyze how many courses with and without a certificate have been completed.

- Manage information about teachers and courses to see who teaches the most courses.

- Monitor what percentage of users successfully complete the courses.

Thus, the OLAP database ensures timely collection, storage and updating of data that is needed to manage the learning process and support analytics.

The main objects that are stored in the DWH database

dwh_Dim_Time

The dwh_Dim_Time table stores calendar data — the date and its division into day, month, quarter, and year. This serves as the basis for analyzing time-based data and allows you to build temporary reports and aggregates.

dwh_Dim_Category

The dwh_Dim_Category table contains course categories that allow you to classify and group courses by topic for ease of analysis and segmentation.

dwh_Dim_Course

The dwh_Dim_Course table stores the main characteristics of the courses: the name, category, and information about the availability of the certificate.

dwh_Dim_User (SCD Type 2)

The dwh_Dim_User table contains detailed data about users of the platform, including the user's unique key (surrogate key), the real user_id from OLTP, name, email, registration date, as well as the periods of validity of records (start_date, end_date) and the relevance flag (is_current). This allows you to keep historical records of changes in user data (SCD — Slowly Changing Dimension).

dwh_Dim_Instructor

The dwh_Dim_Instructor table stores data about teachers: their unique identifier and name.

dwh_Bridge_Instructor_Course

The dwh_Bridge_Instructor_Course table implements a many-to-many relationship between courses and instructors, indicating which teacher teaches which course.

dwh_Fact_Enrollment

The dwh_Fact_Enrollment table records the facts of course completion by users, linking the user, course, and date via dimension keys, as well as the completion status. This table allows you to analyze user activity and progress over time.

dwh_Fact_Payment

The dwh_Fact_Payment table stores data on user payments for time-bound courses. Allows you to perform financial analysis, calculate the amount of income by month and year.

dwh_Fact_Rating

The dwh_Fact_Rating table contains course ratings by users. It is used to analyze the quality and popularity of courses.

Materialized representations are aggregates

dwh_Agg_Monthly_Payments — an aggregated table with payment amounts, the number of paying users, and paid courses by month and year. Simplifies and speeds up the execution of analytical queries on financial indicators.

dwh_Agg_Course_Rating — an aggregate with an average rating and the number of grades for each course, which allows you to quickly evaluate the quality of courses.

Overall description

OLTP — structure and connections

The OLTP database is designed to store operational information on online courses, users and their actions. It contains the following tables:

Main tables:

- **Users** — stores information about users (user_id — PK).
- **Categories** — contains course categories (category_id — PK).

- **Course** —describes the courses. It is linked to Categories by a foreign key (category_id).
- **Records** — records which courses users have taken (PK: user_id + course_id; FK for Users and Course).
- **Rating** — it contains course ratings from users (PK: user_id + course_id; FK for Users and Course; CHECK for a rating from 1 to 5).
- **Certificate** — records the receipt of certificates (FKs for Users and Course).
- **Payment** — course payment information (FKs for Users and Course).
- **Instructors** — stores the names of instructors.
- **Course_Instructors** — a linking table between courses and instructors (PK: course_id + instructor_id).

Temporary tables:

They are used to prepare and import data into the main structure :

- temp_users
- temp_categories_and_courses
- temp_records_and_ratings
- temp_cert_pay
- temp_instructors

Keys and restrictions:

- Unique fields: email in Users, category_name in Categories, course_name in Course, instructor_name in Instructors.
- All foreign keys are linked to parent tables via REFERENCES.

OLAP — structure and connections

The OLAP database is designed to store and analyze aggregated and historical data on online courses, users, their activities, and payments. The structure is based on the "snowflake" principle.

Basic Dimension Tables (Dimensions):

- **dwh_Dim_Time** — stores temporary attributes (time_id — PK): date (unique), day, month, quarter, year.
- **dwh_Dim_Category** — contains course categories (category_id — PK), used for course classification.
- **dwh_Dim_Course** — describes the courses (course_id — PK), includes the name, the foreign key for the category (category_id — FK), as well as information about the availability of the certificate.
- **dwh_Dim_User** — stores user data with a history of changes (user_sk — PK), contains the original user ID (user_id), name, email, registration date, as well as fields for maintaining a history of changes (start_date, end_date, is_current). A unique combination key (user_id, start_date).
- **dwh_Dim_Instructor** — contains information about teachers (instructor_id — PK) and their names.

Connecting table (Bridge):

- **dwh_Bridge_Instructor_Course** — a many-to-many relationship between instructors and courses (PK: instructor_id + course_id), both fields are foreign keys for the corresponding dimensions.

Fact Tables (Facts):

- **dwh_Fact_Enrollment** — the facts of course completion by time-bound users (PK: user_sk + course_id + time_id). Stores the course completion status. All keys are external links to dimensions: user_sk (dwh_Dim_User), course_id (dwh_Dim_Course), time_id (dwh_Dim_Time).
- **dwh_Fact_Payment** — the facts of user payments for time-bound courses (PK: user_sk + course_id + time_id). Contains the payment amount (total_amount). Foreign keys based on user

dimension, course, and time.

- **dwh_Fact_Rating** — the facts of course ratings by users (PK: user_sk + course_id). Stores a rating with a check value from 1 to 5 (CHECK). Foreign keys per user and course.

Keys and restrictions:

- Primary keys (PK) ensure the uniqueness of the records in each table.
- Foreign keys (FKs) link fact tables and linking tables to dimensions, ensuring data integrity.
- Unique restrictions:
 - B dwh_Dim_Time - the unique field is date.
 - B dwh_Dim_User uniqueness is provided for a combination (user_id, start_date) to store the user's change history.
- Data integrity constraints include checking a rating in the range from 1 to 5 in the ratings fact table.

Thus, the OLAP database serves for efficient storage and analytical processing of historical and aggregated data on training, payments, estimates, and users.

Instructions for scripts

Creating an OLTP database structure

1.1 Creating a database “OLTP_OnlineCourses”

1. Open **pgAdmin**.
2. Create a new database named: **OLTP_OnlineCourses**.

1.2 Creating the basic OLTP schema

1. Open the file `create_oltp_schema.sql`.
2. Run the **entire script** in the database OLTP_OnlineCourses.

After the execution, you should have the following **9 tables**:

- 1) Users
- 2) Categories
- 3) Course
- 4) Records
- 5) Rating
- 6) Certificate
- 7) Payment
- 8) Instructors
- 9) Course_Instructors

1.3 Preparing temporary tables and uploading data

1. Open the file `load_csv_to_oltp.sql`.
2. Execute **the part of the script** that contains only **CREATE TABLE** for temporary tables:

- 1) `CREATE TABLE temp_users (...);`
- 2) `CREATE TABLE temp_categories_and_courses (...);`
- 3) `CREATE TABLE temp_records_and_ratings (...);`
- 4) `CREATE TABLE temp_cert_pay (...);`
- 5) `CREATE TABLE temp_instructors (...);`

After completion, **5 more temporary tables** will be added, and the total number of tables will be **14**.

Step 2: Import data from CSV

2.1 Import CSV files into the corresponding temporary tables (all *.csv files are located in the OnlineCourses_scv_files folder)

Go to:

OLTP_OnlineCourses → Schemas → Tables

For each table, follow these steps:

temp_users

Right click → **Import/Export Data**

In the Filename field, specify the file:

Untitled spreadsheet - Users (2).csv

Click **OK**

Then **everything is the same**, but for the rest of the tables :

temp_categories_and_courses

Select a file:

categories_and_courses (2).csv

temp_records_and_ratings

Select a file:

records_and_ratings.csv

temp_cert_pay

Select a file:

certificates_and_payments.csv

temp_instructors

Select a file:

Instructors (2).csv

Step 3: Loading data from temporary tables to main tables

Now, after the data has been loaded into the temporary tables, we can insert them into the main tables.

For this :

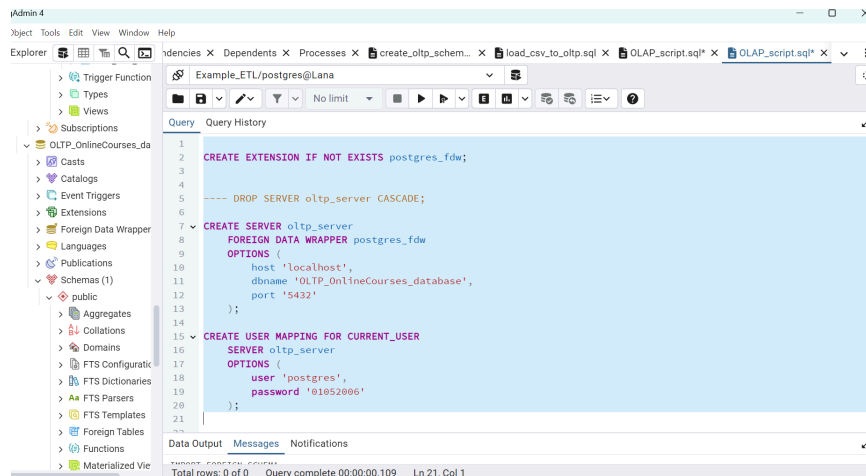
1. Run the rest of the INSERT INTO table_name script.

The data has now been uploaded to the main tables.

Step 4: OLAP and ETL

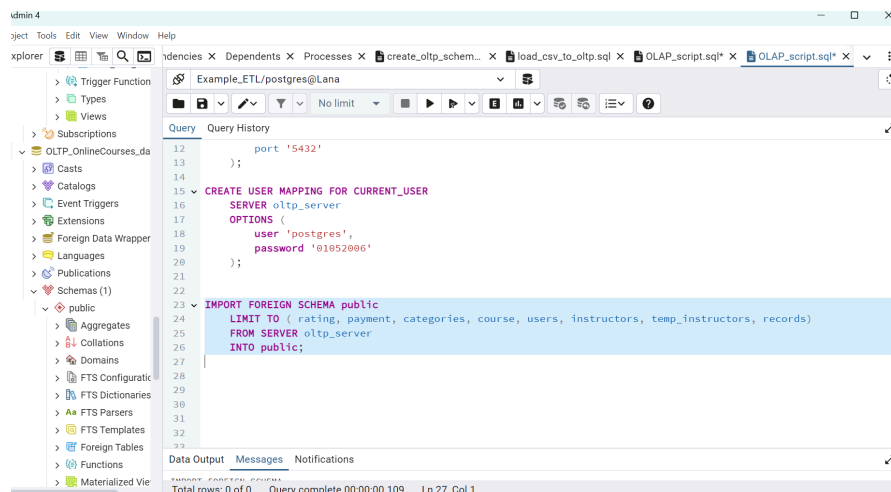
Open the file olap_setup.sql.

In order to connect to the existing OLTP database "OLTP_OnlineCourses" and retrieve the data from there, you need this part of the script :



2. Import tables from the OLTP database

Next, we import the tables, for this we execute this part of the script :



3. Creating and uploading dimensions

Open the file `olap_schema.sql`

And we execute this part of the script :

dwh_Dim_Time

`CREATE TABLE dwh_Dim_Time (...);`

dwh_Dim_Category

```
CREATE TABLE dwh_Dim_Category (...);
```

dwh_Dim_Course

```
CREATE TABLE dwh_Dim_Course (...);
```

dwh_Dim_User (SCD Type 2)

```
CREATE TABLE dwh_Dim_User (...);
```

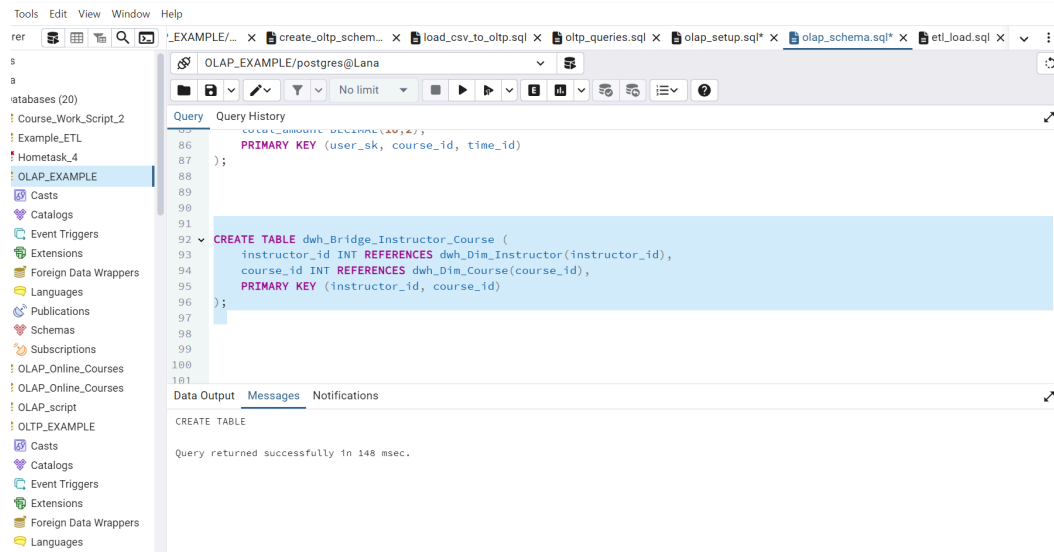
dwh_Dim_Instructor

```
CREATE TABLE dwh_Dim_Instructor (...);
```

4. Creating and loading a bridge table

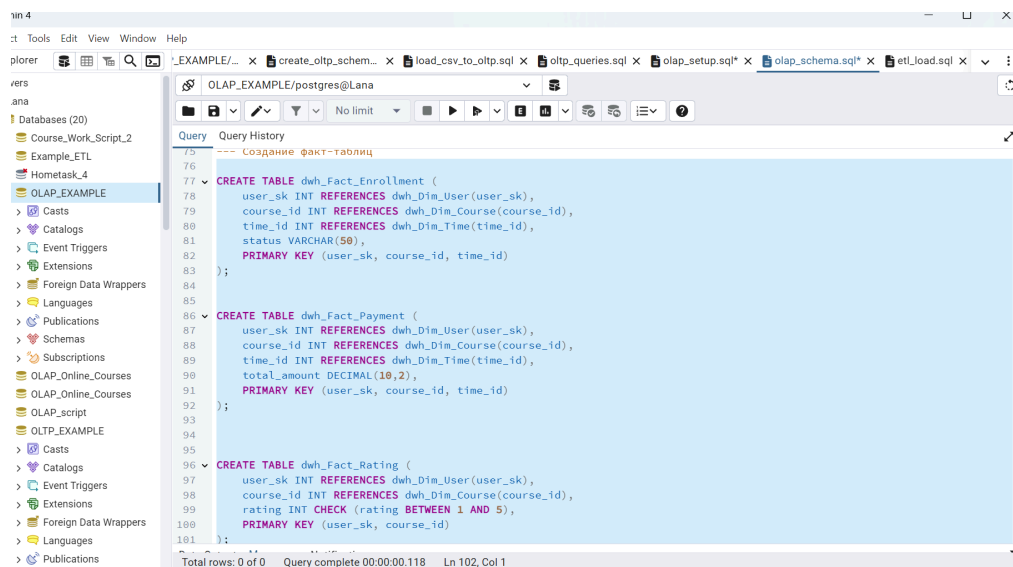
To create a linking table, run this part of the script :

```
dwh_Bridge_Instructor_Course (Connection of courses and instructors)
```



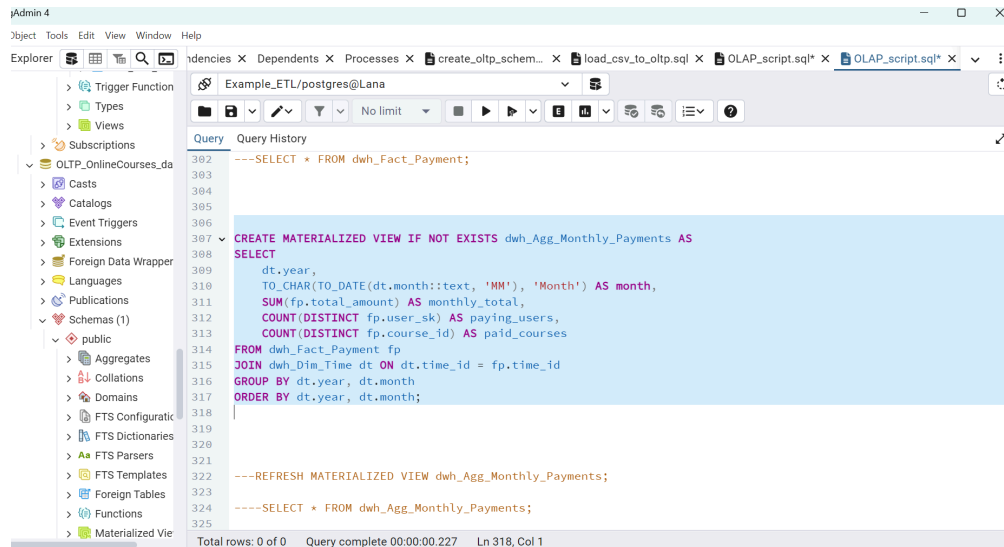
5. Creating fact tables

- 1) **dwh_Fact_Rating** (The fact of the ratings), **dwh_Fact_Enrollment** (The fact of enrolling in courses) and **dwh_Fact_Payment** (The fact of payments)



6. Creating aggregates (materialized views)

1) Monthly payments:



The screenshot shows the SQL Developer interface with a query editor. The left pane displays the database schema, including the 'public' schema and its aggregates. The main query editor contains the following SQL code:

```
---SELECT * FROM dwh_Fact_Payment;

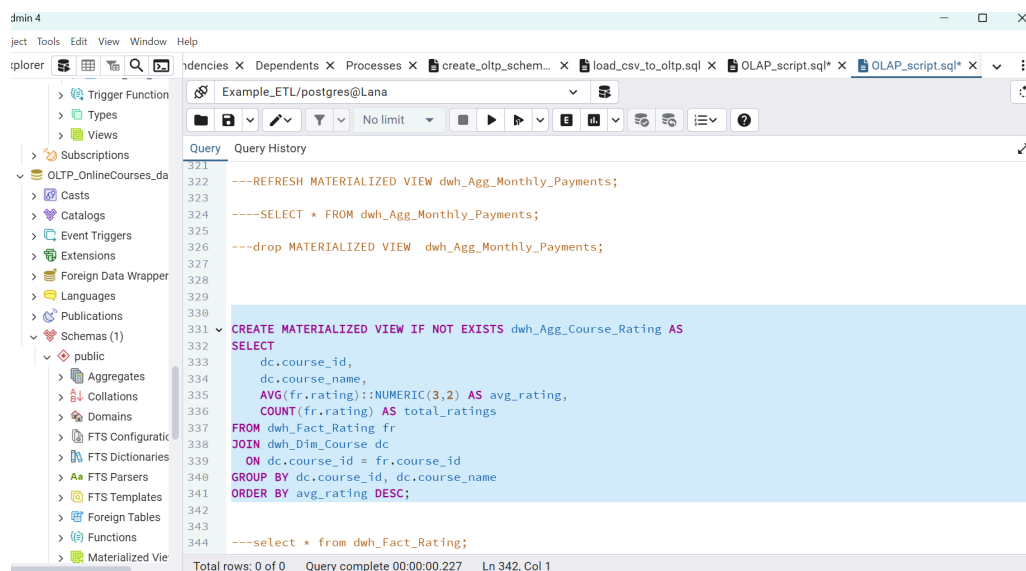
CREATE MATERIALIZED VIEW IF NOT EXISTS dwh_Agg_Monthly_Payments AS
SELECT
    dt.year,
    TO_CHAR(TO_DATE(dt.month::text, 'MM'), 'Month') AS month,
    SUM(fp.total_amount) AS monthly_total,
    COUNT(DISTINCT fp.user_sk) AS paying_users,
    COUNT(DISTINCT fp.course_id) AS paid_courses
FROM dwh_Fact_Payment fp
JOIN dwh_Dim_Time dt ON dt.time_id = fp.time_id
GROUP BY dt.year, dt.month
ORDER BY dt.year, dt.month;

---REFRESH MATERIALIZED VIEW dwh_Agg_Monthly_Payments;

----SELECT * FROM dwh_Agg_Monthly_Payments;
```

The status bar at the bottom indicates: Total rows: 0 of 0 Query complete 00:00:00.227 Ln 318, Col 1.

2) Average course grade :



The screenshot shows the SQL Developer interface with a query editor. The left pane displays the database schema, including the 'public' schema and its aggregates. The main query editor contains the following SQL code:

```
---REFRESH MATERIALIZED VIEW dwh_Agg_Monthly_Payments;

----SELECT * FROM dwh_Agg_Monthly_Payments;

---drop MATERIALIZED VIEW dwh_Agg_Monthly_Payments;

CREATE MATERIALIZED VIEW IF NOT EXISTS dwh_Agg_Course_Rating AS
SELECT
    dc.course_id,
    dc.course_name,
    AVG(fr.rating)::NUMERIC(3,2) AS avg_rating,
    COUNT(fr.rating) AS total_ratings
FROM dwh_Fact_Rating fr
JOIN dwh_Dim_Course dc
    ON dc.course_id = fr.course_id
GROUP BY dc.course_id, dc.course_name
ORDER BY avg_rating DESC;

---select * from dwh_Fact_Rating;
```

The status bar at the bottom indicates: Total rows: 0 of 0 Query complete 00:00:00.227 Ln 342, Col 1.

Step 5: Download the data

- 1) Open the file `etl_load.sql`

Uploading data to **Dimension Tables**

To do this, run this part of the script :

INSERT INTO dwh_Dim_Time — dates from the Payment table, with extraction of the day, month, quarter, and year.

INSERT INTO dwh_Dim_Category — unique categories from the Categories table.

INSERT INTO dwh_Dim_Course — courses linked to categories.

INSERT INTO dwh_Dim_User — users with SCD Type 2 support (tracking changes by name and email fields).

INSERT INTO dwh_Dim_Instructor — instructors.

INSERT INTO dwh_Bridge_Instructor_Course — a bridge table between courses and teachers.

All inserts were made only for new records using **NOT EXISTS** or **NOT IN** to avoid duplication.

Next, upload the data to the **Fact Tables**

INSERT INTO dwh_Fact_Rating

INSERT INTO dwh_Fact_Enrollment

INSERT INTO dwh_Fact_Payment

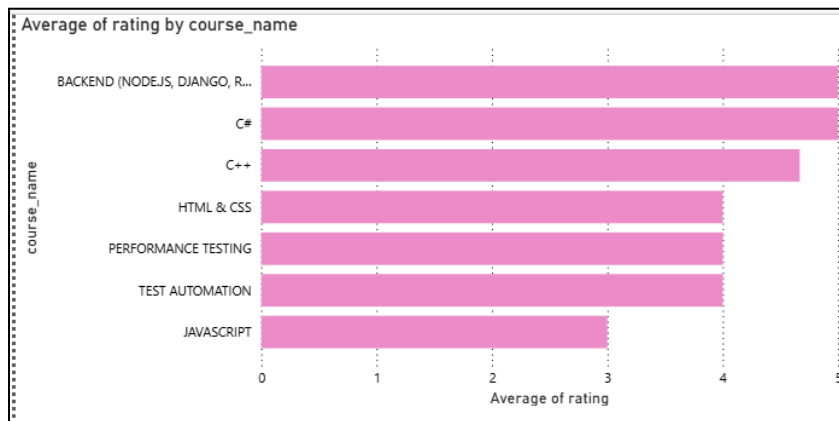
When inserting into the fact tables, **JOINS** were applied to the dimensions and checking for the absence of duplicate records (**NOT EXISTS**).

DISTINCT was also used in certain places to avoid mistakes when violating unique restrictions.

Power BI Report

On page 1 we can see an analysis of online course sales.

1. Course rating (Bar Chart)



What the graph shows :

There is a bar chart which describes the trend of courses by average rating.

How it reads :

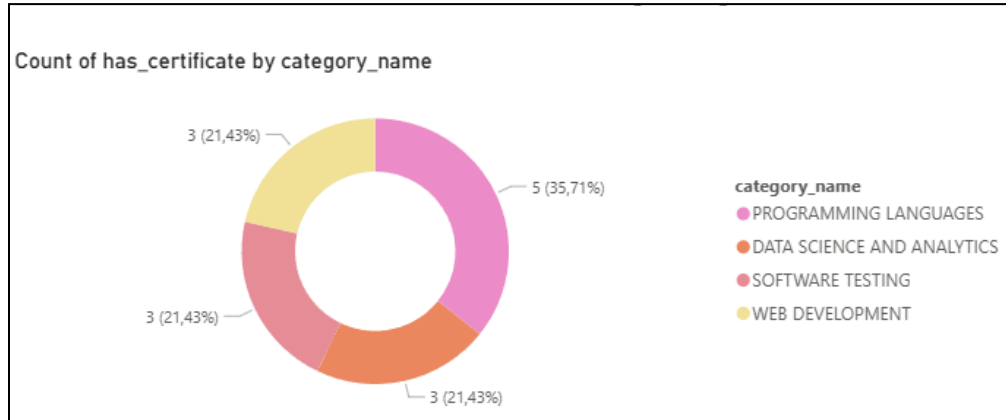
On the Y—axis are the names of the courses (vertically).

On the X— average rating (horizontally).

Description of the chart :

Courses such as Backend, C++, C# have the highest rating of about 5. And JavaScript has the lowest rating and all other courses have a rating from 3 to 5.

2. Number of certificates by category (Pie Chart)



What the graph shows :

There is a pie chart which describes the number of certified courses by category.

How it reads :

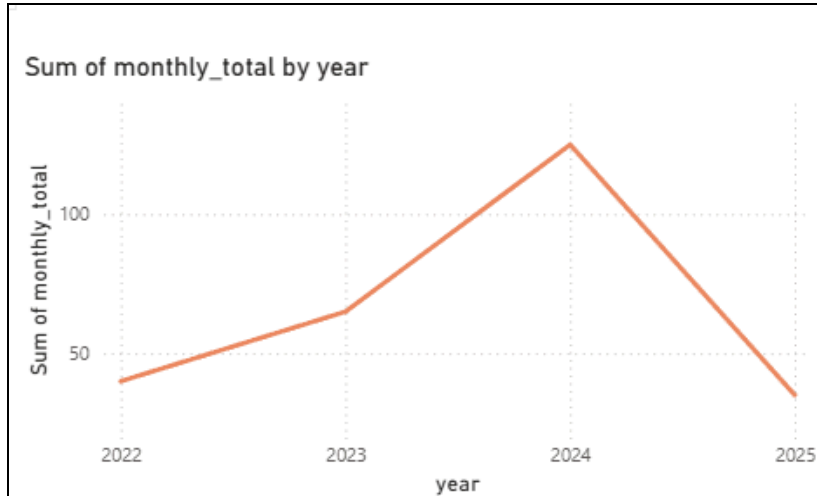
Each sector is a category of courses (for example "Programming languages", "Web development " and other categories).

And the number around the sector indicates how many certificates were obtained in this category of courses.

Description of the chart :

We can see that major part of certificates obtained category "Programming languages" has 5 certificates per category and last 3 categories obtained equal number of courses. They have 3 certificates per category.

3. Dynamics of course purchases by year (Line Chart)



What the graph shows :

There is a line chart which describes the annual trend in course purchases over multiple years.

How it reads :

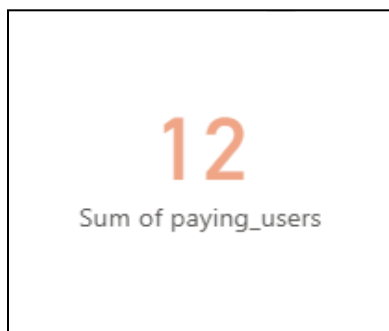
On the Y— this is the income for each year.

On the X— axis are the years (2022, 2023, 2024, 2025).

Description of the chart :

We can see that from 2022 to 2024 the revenue dynamics increased a lot, but in 2025 it is much lower, but now it is only the beginning of the year and accordingly we do not have all the data for the year

4. Sum of users on our platform (Card visual)



This card describes how many users are on our platform.

5. Slicers

First slicer :

Allows you to select one or more years.

After the selection, the data is filtered so that only the data for the selected years is displayed.

Second slicer :

Allows you to select one or more course categories

After the selection, the data is displayed only for the selected categories.