SPRAWOZDANIE

Projekt zaliczeniowy

KAROL ZACHAREWICZ
GRUPA 2

ADAM WEŁNICKI
GRUPA 2

ZAAWANSOWANE SYSTEMY BAZ DANYCH

SPIS TREŚCI

ŚRODOWISKO

ORACLE LINUX 8.8

Obraz systemu: OracleLinux-R8-U8-x86_64-dvd.iso

Typ instalacji minimalny, konsolowy bez GUI.

Pliki instalacyjne bazy danych, APEX i Oracle REST Data Services:

Oracle XE	oracle-database-xe-21c-1.0-1.ol8.x86_64
APEX	apex_24.2_en.zip
ORDS	ords-24.4.0.345.1601.zip

Po zainstalowaniu systemu instalujemy potrzebne pakiety:

```
sudo dnf install -y vim nano wget curl tar unzip zip net-tools lsof NetworkManager NetworkManager-tui
```

Następnie konfigurujemy ustawienia sieciowe:

```
nmtui
```

Tworzymy (jako root) grupę i użytkownika oracle, dodajemy go do grupy root (wheel).

```
sudo groupadd oinstall
sudo groupadd dba
sudo useradd -m -g oinstall -G dba -s /bin/bash oracle
sudo passwd oracle
usermod -aG wheel oracle
```

ORACLE XE

Instalujemy bazę danych Oracle XE

```
cd /home/downloads   # ścieżka z plikami
sudo dnf install -y ./oracle-database-xe-21c-1.0-1.ol8.x86_64.rpm
```

Uruchomiamy konfiguracje:

```
sudo /etc/init.d/oracle-xe-21c configure
```

Po konfiguracji przełączamy się na użytwkoniak oracle i konfigurujemy środiwsko:

```
su – oracle
nano ~/.bash_profile
```

Dodaj na końcu pliku:

```
export ORACLE_SID=XE
export ORAENV_ASK=NO
. /opt/oracle/product/21c/dbhomeXE/bin/oraenv
export PATH=$PATH:$ORACLE_HOME/bin
```

Odświeżamy plik

```
source ~/.bash_profile
```

Dodajemy reguły do firewalle (od razu dla wszystkich usług)

```
sudo firewall-cmd --permanent --add-port=1521/tcp
sudo firewall-cmd --permanent --add-port=8080/tcp
sudo firewall-cmd --permanent --add-port=8443/tcp
sudo firewall-cmd --reload
```

Możemy już sprawdzić czy można się połaczyć z bazą danych, przez użytkwonika systemowego SYS, np. w SQL Develop:

Przechodzimy do bazy danych w celu utworzenia nowego użytkownika i stworzenia idywidualnego tablespace, tak, aby odizolować tabele systemowe od naszych.

```
# jako użytkownik oracle
sqlplus sys as sysdba
ALTER SESSION SET CONTAINER = XEPDB1;

CREATE TABLESPACE karol_tbs
    DATAFILE '/opt/oracle/oradata/XE/XEPDB1/karol_tbs01.dbf'
    SIZE 100M
    AUTOEXTEND ON
    NEXT 50M
    MAXSIZE UNLIMITED;

CREATE USER karol IDENTIFIED BY mozilla511
    DEFAULT TABLESPACE karol_tbs
    TEMPORARY TABLESPACE TEMP
    QUOTA UNLIMITED ON karol_tbs;

GRANT CONNECT TO karol;
GRANT RESOURCE TO karol;
GRANT CREATE SESSION TO karol;
GRANT CREATE TABLE TO karol;
GRANT CREATE VIEW TO karol;
```

```
GRANT CREATE SEQUENCE TO karol;

GRANT CREATE SYNONYM TO karol;

GRANT CREATE PROCEDURE TO karol;

GRANT CREATE FUNCTION TO karol;

GRANT CREATE PACKAGE TO karol;

GRANT CREATE TRIGGER TO karol;

GRANT CREATE TYPE TO karol;

GRANT CREATE MATERIALIZED VIEW TO karol;

GRANT CREATE JOB TO karol;

GRANT EXECUTE ON DBMS_SCHEDULER TO karol;

GRANT SELECT_CATALOG_ROLE TO karol;

GRANT EXECUTE ANY PROCEDURE TO karol;

GRANT UNLIMITED TABLESPACE TO karol;
```
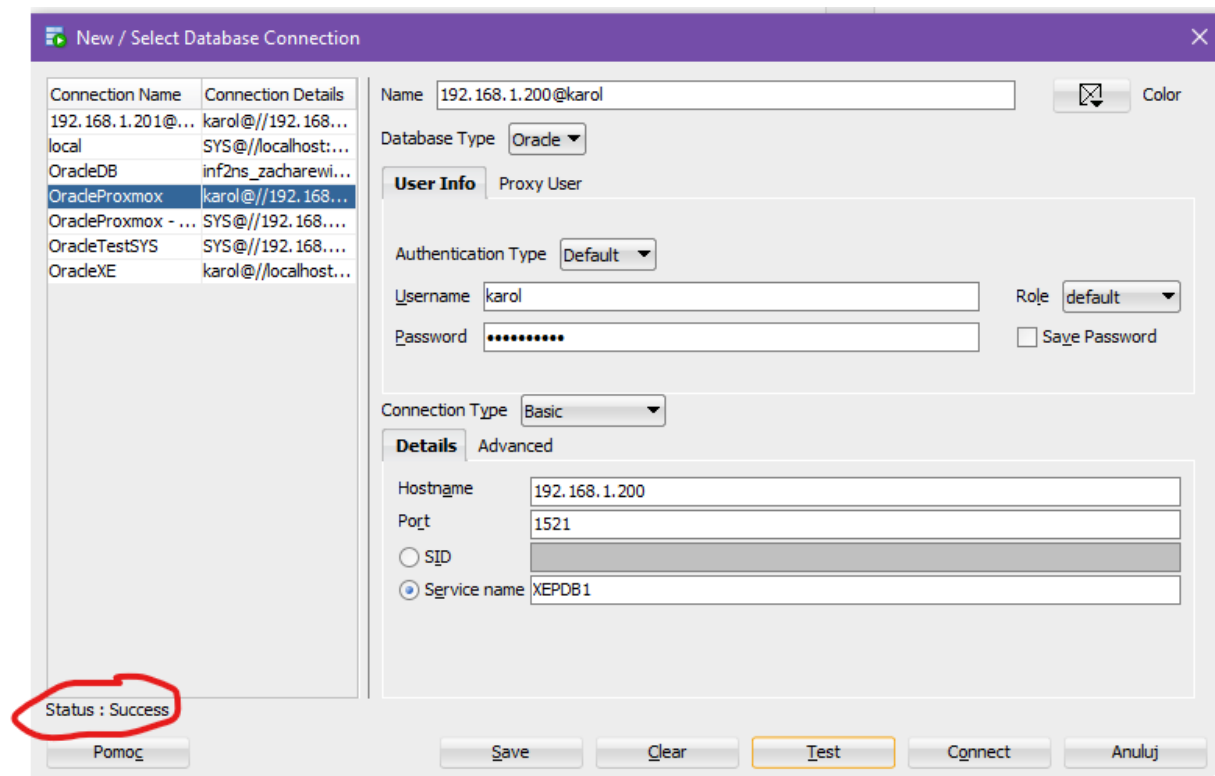
Po utworzeniu I nadaniu uprawnień sprawdzamy czy możemy się połączyć.

APEX I ORDS

Przechodzimy do instalacji APEX, jako użytkownik oracle

```
su - oracle
cd /opt/oracle
unzip /home/downloads/apex_24.2_en.zip
```

```
cd /opt/oracle/apex
sqlplus sys as sysdba


ALTER SESSION SET CONTAINER = XEPDB1;


@apexins.sql SYSAUX SYSAUX TEMP /i/
@apxchpwd.sql
UWAGA! Hasło poddawane jest walidacji więc musi spełniać wymogi Oracle.
```

```
SYS> @apxchpwd.sql
...set_appun.sql
================================================================================
This script can be used to change the password of an Oracle APEX
instance administrator. If the user does not yet exist, a user record will be
created.
================================================================================
Enter the administrator's username [ADMIN]
User "ADMIN" does not yet exist and will be created.
Enter ADMIN's email [ADMIN]
Enter ADMIN's password []
Created instance administrator ADMIN.
```

```
exit
```

Do prawidłowego działania APEX i ORDS potrzebujemy odpowiedniej wersji Java.

```
sudo dnf install java-17-openjdk java-17-openjdk-devel -y
```

Nastęnie przechodzimy do instalacji ORDS:

```
cd /opt/oracle
unzip /home/downloads/ords-24.4.0.345.1601.zip -d ords
cd ords
mkdir /opt/oracle/ords-config
lsnrctl status # sprawdź czy nasłuchuje
java -jar ords.war --config /opt/oracle/ords-config install
```

```
Oracle REST Data Services - Interactive Install

  Enter a number to select the database connection type to use
    [1] Basic (host name, port, service name)
    [2] TNS (TNS alias, TNS directory)
    [3] Custom database URL
  Choose [1]: 1
  Enter the database host name [localhost]: localhost
  Enter the database listen port [1521]: 1521
  Enter the database service name [XE]: XEPDB1
  Provide database user name with administrator privileges.
    Enter the administrator username: SYS
  Enter the database password for SYS AS SYSDBA:
```

```
  Enter a number to update the value or select option A to Accept and Continue
    [1] Connection Type: Basic
    [2] Basic Connection: HOST=localhost PORT=1521 SERVICE_NAME=XEPDB1
        Administrator User: SYS AS SYSDBA
    [3] Database password for ORDS runtime user (ORDS_PUBLIC_USER): <generate>
    [4] ORDS runtime user and schema tablespaces:  Default: SYSAUX Temporary TEMP
    [5] Additional Feature: Database Actions
    [6] Configure and start ORDS in Standalone Mode: Yes
    [7]    Protocol: HTTP
    [8]       HTTP Port: 8080
    [9]    APEX static resources location:
    [A] Accept and Continue - Create configuration and Install ORDS in the database
    [Q] Quit - Do not proceed. No changes
  Choose [A]: A
```

W razie problemów należy sprawdzić czy ords nasłuchuje na odpowiednim porcie i czy wczytuje kontener XEPDB1.

```
lsnrctl status


Service "xepdb1" has 1 instance(s).

 Instance "XE", status READY, has 1 handler(s) for this service...
```

Dodajemy do zapory, jeżeli nie zrobiliśmy tego wcześniej:

```
sudo firewall-cmd --add-port=8080/tcp --permanent

sudo firewall-cmd –reload
```

Nalezy jeszcze odblokować użytkowników APEX:

```
sqlplus sys as sysdba


ALTER SESSION SET CONTAINER = XEPDB1;

ALTER USER APEX_PUBLIC_USER ACCOUNT UNLOCK;

ALTER USER APEX_PUBLIC_USER IDENTIFIED BY mozilla511;
```

```
ALTER USER ORDS_PUBLIC_USER ACCOUNT UNLOCK;

ALTER USER ORDS_PUBLIC_USER IDENTIFIED BY mozilla511;

exit
```

Ustawiamy hasło

```
java -jar ords.war --config /opt/oracle/ords-config config secret db.password
```

Następnie

```
mkdir -p /opt/oracle/ords/i

ln -s /opt/oracle/apex/images /opt/oracle/ords/i


java -jar ords.war --config /opt/oracle/ords-config config set standalone.static.path /opt/oracle/ords/i


mkdir -p /opt/oracle/ords/i

cp -r /opt/oracle/apex/images/* /opt/oracle/ords/i/


cd /opt/oracle/ords

java -jar ords.war --config /opt/oracle/ords-config serve
```

```
[oracle@localhost ~]$ cd /opt/oracle/ords
[oracle@localhost ords]$ java -jar ords.war --config /opt/oracle/ords-config serve
Warning: Support for executing: java -jar ords.war has been deprecated.
Please add ords to your PATH and use the ords command instead.
Run the following command to add ords to your PATH:

echo -e 'export PATH="$PATH:/opt/oracle/ords/bin"' >> ~/.bash_profile

Start a new shell to pick up this change.

ORDS: Release 24.4 Production on pt. kwi 11 10:10:13 2025

Copyright (c) 2010, 2025, Oracle.

Configuration:
  /opt/oracle/ords-config

2025-04-11T08:10:14.163Z INFO        HTTP and HTTP/2 cleartext listening on host: 0.0.
2025-04-11T08:10:14.205Z INFO        Disabling document root because the specified fol
2025-04-11T08:10:14.207Z INFO        Default forwarding from / to contextRoot configur
2025-04-11T08:10:21.065Z INFO        Configuration properties for: |default|lo|
db.servicename=XEPDB1
```

Teraz możemy wejść na stronę i się zalogować.

```
http://192.168.1.200:8080/ords/
```
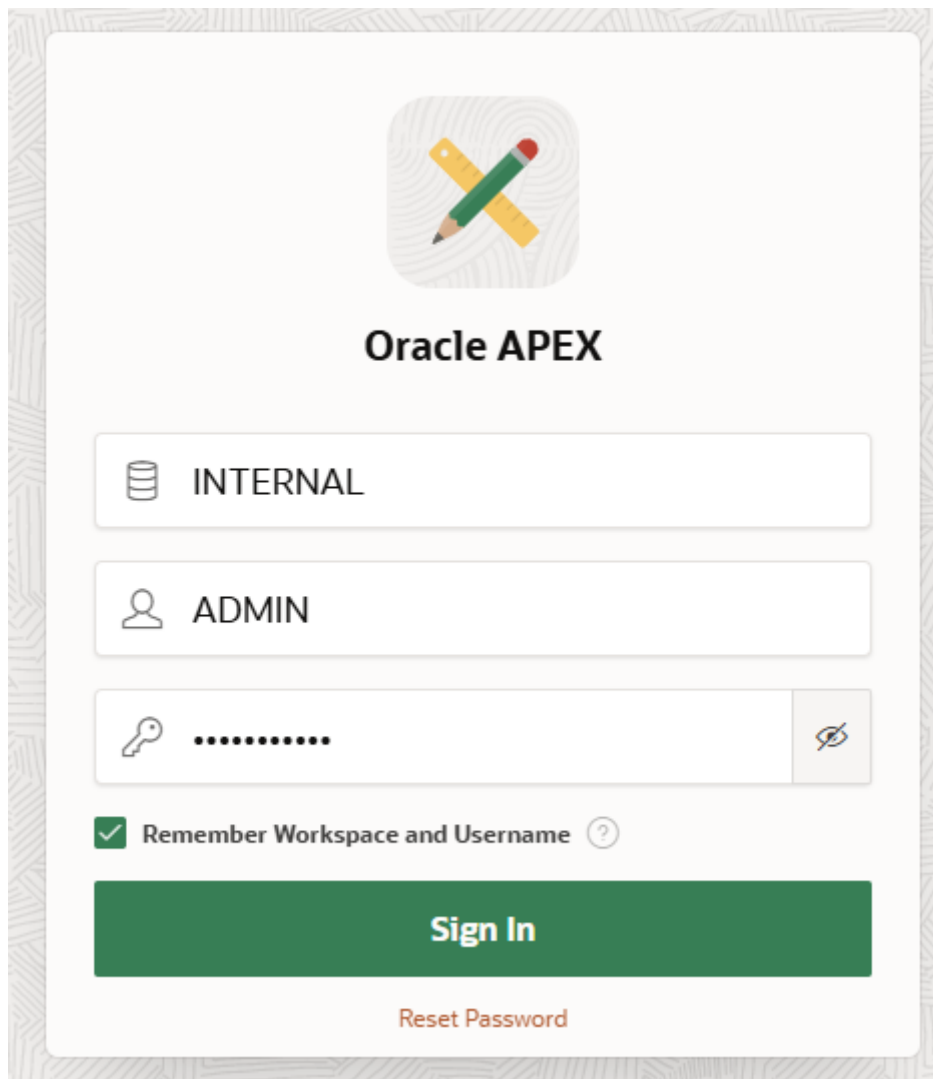
FLASK

Flask będzie używany do używania skryptów automatyzujących pobieranie danych jak również do zwracania zapytań wykorzystywanych w APEX.

Na początek instalujemy Python'a. Z uwagi, że serwer jest przeznaczony tylko do projektu nie tworzymy wirtualnego środwiska tylko pracujemy na systemowym.

```
dnf install -y python38
# Jeżeli brak w repo
dnf install -y oracle-epel-release-el8
dnf update -y


dnf install -y python38
ln -s /usr/bin/python3.8 /usr/bin/python
ln -s /usr/bin/pip3.8 /usr/bin/pip
```

```
pip install Flask yfinance oracledb

sudo firewall-cmd --add-port=5001/tcp --permanent
sudo firewall-cmd --reload
/usr/bin/python3.8 /home/oracle/stock_oracle/scripts/test_flask.py
# Jezeli dodaliśmy alias:
python /home/oracle/stock_oracle/scripts/test_flask.py
```

Na koniec ustawiamy pliki ervice w celu obsługi przez systemctl:

sudo nano /etc/systemd/system/ords.service

```
[Unit]
Description=Oracle REST Data Services
After=network.target

[Service]
User=oracle
Group=oinstall
WorkingDirectory=/opt/oracle/ords
ExecStart=/usr/bin/java -jar /opt/oracle/ords/ords.war --config /opt/oracle/ords-config serve
Restart=on-failure
SuccessExitStatus=143

[Install]
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload
sudo systemctl enable ords
```

sudo nano /etc/systemd/system/flask.service

```
sudo nano /etc/systemd/system/flask.service
[Unit]
Description=Flask Application Service
After=network.target
```

```
[Service]
User=oracle
Group= oinstall
WorkingDirectory=/home/oracle/stock_oracle/scripts
ExecStart=/usr/bin/python3.8 /home/oracle/stock_oracle/scripts/test_flask.py
Restart=on-failure


[Install]
WantedBy=multi-user.target
```

```
sudo systemctl daemon-reload
sudo systemctl enable flask
```

sudo nano /etc/systemd/system/oracle-xe.service

```
[Unit]
Description=Alias for Oracle XE 21c
After=network.target

[Service]
Type=forking
ExecStart=/etc/init.d/oracle-xe-21c start
ExecStop=/etc/init.d/oracle-xe-21c stop
ExecReload=/etc/init.d/oracle-xe-21c restart
RemainAfterExit=true

[Install]
WantedBy=multi-user.target
```
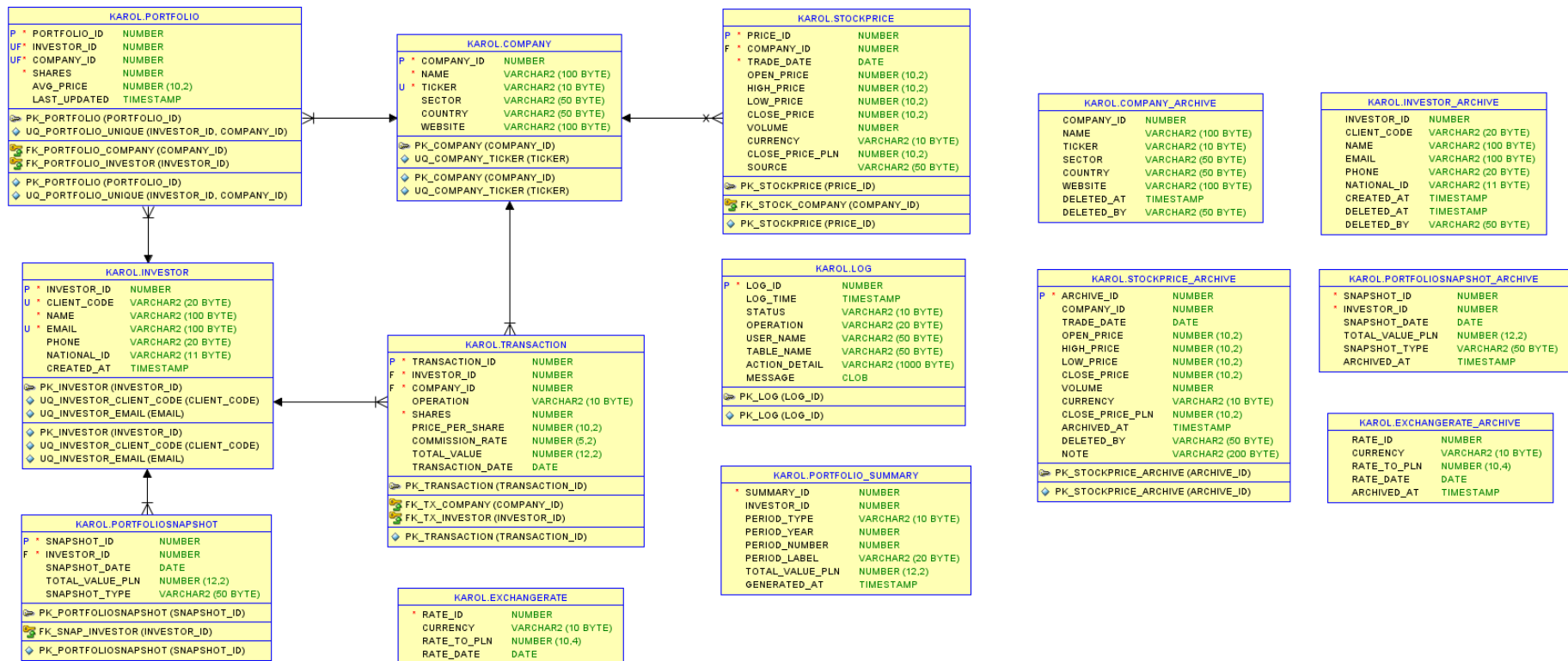
```
sudo systemctl daemon-reload
sudo systemctl enable oracle-xe
```

SCHEMAT BAZY DANYCH

KAROL.PORTFOLIO

P	*	PORTFOLIO_ID	NUMBER
UF	*	INVESTOR_ID	NUMBER
UF	*	COMPANY_ID	NUMBER
	*	SHARES	NUMBER
		AVG_PRICE	NUMBER (10,2)
		LAST_UPDATED	TIMESTAMP

PK_PORTFOLIO (PORTFOLIO_ID)
UQ_PORTFOLIO_UNIQUE (INVESTOR_ID, COMPANY_ID)

FK_PORTFOLIO_COMPANY (COMPANY_ID)
FK_PORTFOLIO_INVESTOR (INVESTOR_ID)

PK_PORTFOLIO (PORTFOLIO_ID)
UQ_PORTFOLIO_UNIQUE (INVESTOR_ID, COMPANY_ID)

KAROL.COMPANY

P	*	COMPANY_ID	NUMBER
	*	NAME	VARCHAR2 (100 BYTE)
U	*	TICKER	VARCHAR2 (10 BYTE)
		SECTOR	VARCHAR2 (50 BYTE)
		COUNTRY	VARCHAR2 (50 BYTE)
		WEBSITE	VARCHAR2 (100 BYTE)

PK_COMPANY (COMPANY_ID)
UQ_COMPANY_TICKER (TICKER)

PK_COMPANY (COMPANY_ID)
UQ_COMPANY_TICKER (TICKER)

KAROL.STOCKPRICE

P	*	PRICE_ID	NUMBER
F	*	COMPANY_ID	NUMBER
	*	TRADE_DATE	DATE
		OPEN_PRICE	NUMBER (10,2)
		HIGH_PRICE	NUMBER (10,2)
		LOW_PRICE	NUMBER (10,2)
		CLOSE_PRICE	NUMBER (10,2)
		VOLUME	NUMBER
		CURRENCY	VARCHAR2 (10 BYTE)
		CLOSE_PRICE_PLN	NUMBER (10,2)
		SOURCE	VARCHAR2 (50 BYTE)

PK_STOCKPRICE (PRICE_ID)

FK_STOCK_COMPANY (COMPANY_ID)

PK_STOCKPRICE (PRICE_ID)

KAROL.COMPANY_ARCHIVE

COMPANY_ID	NUMBER
NAME	VARCHAR2 (100 BYTE)
TICKER	VARCHAR2 (10 BYTE)
SECTOR	VARCHAR2 (50 BYTE)
COUNTRY	VARCHAR2 (50 BYTE)
WEBSITE	VARCHAR2 (100 BYTE)
DELETED_AT	TIMESTAMP
DELETED_BY	VARCHAR2 (50 BYTE)

KAROL.INVESTOR_ARCHIVE

INVESTOR_ID	NUMBER
CLIENT_CODE	VARCHAR2 (20 BYTE)
NAME	VARCHAR2 (100 BYTE)
EMAIL	VARCHAR2 (100 BYTE)
PHONE	VARCHAR2 (20 BYTE)
NATIONAL_ID	VARCHAR2 (11 BYTE)
CREATED_AT	TIMESTAMP
DELETED_AT	TIMESTAMP
DELETED_BY	VARCHAR2 (50 BYTE)

KAROL.INVESTOR

P	*	INVESTOR_ID	NUMBER
U	*	CLIENT_CODE	VARCHAR2 (20 BYTE)
	*	NAME	VARCHAR2 (100 BYTE)
U	*	EMAIL	VARCHAR2 (100 BYTE)
		PHONE	VARCHAR2 (20 BYTE)
		NATIONAL_ID	VARCHAR2 (11 BYTE)
		CREATED_AT	TIMESTAMP

PK_INVESTOR (INVESTOR_ID)
UQ_INVESTOR_CLIENT_CODE (CLIENT_CODE)
UQ_INVESTOR_EMAIL (EMAIL)

PK_INVESTOR (INVESTOR_ID)
UQ_INVESTOR_CLIENT_CODE (CLIENT_CODE)
UQ_INVESTOR_EMAIL (EMAIL)

KAROL.LOG

P	*	LOG_ID	NUMBER
		LOG_TIME	TIMESTAMP
		STATUS	VARCHAR2 (10 BYTE)
		OPERATION	VARCHAR2 (20 BYTE)
		USER_NAME	VARCHAR2 (50 BYTE)
		TABLE_NAME	VARCHAR2 (50 BYTE)
		ACTION_DETAIL	VARCHAR2 (1000 BYTE)
		MESSAGE	CLOB

PK_LOG (LOG_ID)

PK_LOG (LOG_ID)

KAROL.TRANSACTION

P	*	TRANSACTION_ID	NUMBER
F	*	INVESTOR_ID	NUMBER
F	*	COMPANY_ID	NUMBER
		OPERATION	VARCHAR2 (10 BYTE)
	*	SHARES	NUMBER
		PRICE_PER_SHARE	NUMBER (10,2)
		COMMISSION_RATE	NUMBER (5,2)
		TOTAL_VALUE	NUMBER (12,2)
		TRANSACTION_DATE	DATE

PK_TRANSACTION (TRANSACTION_ID)

FK_TX_COMPANY (COMPANY_ID)
FK_TX_INVESTOR (INVESTOR_ID)

PK_TRANSACTION (TRANSACTION_ID)

KAROL.STOCKPRICE_ARCHIVE

P	*	ARCHIVE_ID	NUMBER
		COMPANY_ID	NUMBER
		TRADE_DATE	DATE
		OPEN_PRICE	NUMBER (10,2)
		HIGH_PRICE	NUMBER (10,2)
		LOW_PRICE	NUMBER (10,2)
		CLOSE_PRICE	NUMBER (10,2)
		VOLUME	NUMBER
		CURRENCY	VARCHAR2 (10 BYTE)
		CLOSE_PRICE_PLN	NUMBER (10,2)
		ARCHIVED_AT	TIMESTAMP
		DELETED_BY	VARCHAR2 (50 BYTE)
		NOTE	VARCHAR2 (200 BYTE)

PK_STOCKPRICE_ARCHIVE (ARCHIVE_ID)

PK_STOCKPRICE_ARCHIVE (ARCHIVE_ID)

KAROL.PORTFOLIOSNAPSHOT_ARCHIVE

	*	SNAPSHOT_ID	NUMBER
	*	INVESTOR_ID	NUMBER
		SNAPSHOT_DATE	DATE
		TOTAL_VALUE_PLN	NUMBER (12,2)
		SNAPSHOT_TYPE	VARCHAR2 (50 BYTE)
		ARCHIVED_AT	TIMESTAMP

KAROL.EXCHANGERATE_ARCHIVE

RATE_ID	NUMBER
CURRENCY	VARCHAR2 (10 BYTE)
RATE_TO_PLN	NUMBER (10,4)
RATE_DATE	DATE
ARCHIVED_AT	TIMESTAMP

KAROL.PORTFOLIOSNAPSHOT

P	*	SNAPSHOT_ID	NUMBER
F	*	INVESTOR_ID	NUMBER
		SNAPSHOT_DATE	DATE
		TOTAL_VALUE_PLN	NUMBER (12,2)
		SNAPSHOT_TYPE	VARCHAR2 (50 BYTE)

PK_PORTFOLIOSNAPSHOT (SNAPSHOT_ID)

FK_SNAP_INVESTOR (INVESTOR_ID)

PK_PORTFOLIOSNAPSHOT (SNAPSHOT_ID)

KAROL.PORTFOLIO_SUMMARY

	*	SUMMARY_ID	NUMBER
		INVESTOR_ID	NUMBER
		PERIOD_TYPE	VARCHAR2 (10 BYTE)
		PERIOD_YEAR	NUMBER
		PERIOD_NUMBER	NUMBER
		PERIOD_LABEL	VARCHAR2 (20 BYTE)
		TOTAL_VALUE_PLN	NUMBER (12,2)
		GENERATED_AT	TIMESTAMP

KAROL.EXCHANGERATE

	*	RATE_ID	NUMBER
		CURRENCY	VARCHAR2 (10 BYTE)
		RATE_TO_PLN	NUMBER (10,4)
		RATE_DATE	DATE

Tabela	Opis
COMPANY	Przechowuje podstawowe informacje o firmach takie jak identyfikator, nazwa, ticker, sektor, kraj i strona internetowa.
COMPANY_ARCHIVE	Archiwizuje rekordy z tabeli COMPANY
INVESTOR	Zawiera dane inwestorów – identyfikator, kod klienta, imię i nazwisko, adres e-mail, numer telefonu, numer identyfikacyjny oraz datę utworzenia, służąc do zarządzania danymi klientów inwestycyjnych
INVESTOR_ARCHIVE	Przechowuje archiwalne dane o inwestorach.
EXCHANGERATE	Zawiera informacje o kursach walut (konwersja z podanej waluty do PLN) dla poszczególnych dat, które mogą być wykorzystane przy przeliczaniu wartości akcji i portfeli.
EXCHANGERATE_ARCHIVE	Archiwizuje rekordy z tabeli EXCHANGERATE
LOG	Rejestruje operacje systemowe (takie jak insert, update czy delete) oraz błędy występujące podczas operacji.
STOCKPRICE	Przechowuje dane notowań akcji firm (otwarcie, maksimum, minimum, zamknięcie, wolumen oraz kurs w PLN uzyskany na podstawie kursu waluty).
STOCKPRICE_ARCHIVE	Archiwizuje dane notowań akcji.
PORTFOLIO	Rejestruje bieżące posiadania inwestorów – ilość akcji posiadanych dla każdej firmy wraz z ceną średnią zakupu i datą ostatniej aktualizacji; stanowi podstawę do wyliczania bieżącej wartości portfela
TRANSACTION	Zawiera szczegółowy zapis transakcji kupna i sprzedaży akcji, rejestrując typ operacji, liczbę akcji, cenę za akcję, prowizję, łączną wartość transakcji oraz datę operacji, co pozwala śledzić historię transakcji inwestorów.
PORTFOLIOSNAPSHOT	Rejestruje momentalne migawki wartości portfela inwestora, zwykle wyliczane poprzez zestawienie danych z tabeli PORTFOLIO z aktualnymi notowaniami akcji
PORTFOLIOSNAPSHOT_ARCHIVE	Archiwizuje wcześniejsze migawki portfela.
PORTFOLIO_SUMMARY	Służy do tworzenia zagregowanych zestawień wartości portfeli za dany okres (np. miesięczny, kwartalny roczny) dla poszczególnych inwestorów.

TWORZENIE TABEL:

sql/schema/1_create.sql

Wypełnienie tabel danymi:

sql/schema/2_insert.sql

DROP USER test9 CASCADE;

SELECT username, account_status, default_tablespace, temporary_tablespace

FROM dba_users

WHERE oracle_maintained = 'N'

ORDER BY username;

W celu utworzenia tabel w bazie danych należy wywołać skrypt:

```
./create_and_init_user.sh nazwa_uzytkownika haslo
```

Zostanie utworzony użytkownik z tablespaces i będzie można się zalogować:

```
Użytkownik test12 został utworzony i baza zainicjalizowana.
----------------------------------------
Host:              192.168.1.200
Port:              1521
Service Name:      XEPDB1
Username:          test12
Password:          mozilla511
----------------------------------------
```

SKRYPT ŁADUJĄCY DANE DO BAZY

- Może pobierać dane ze strony – np. odpalany co jakiś czas/ładowanie z plików csv, json. (deamon, cron table, itd)
- Sprawdzanie poprawności danych
- Gromadzenie i archiwizacja przetworzonych/załadowanych danych do bazy

SKRYPT AKTUALIZUJĄCY STOCKPRICE:

Plik sh, uruchamiający skrypt:

```bash
#!/bin/bash

TIMESTAMP=$(date +"%d%m%Y_%H%M")

LOG_DIR="/home/oracle/stock_oracle/logs"
LOG_FILE="${LOG_DIR}/cron_stock_update_${TIMESTAMP}.log"

mkdir -p "$LOG_DIR"
```

```
cd /home/oracle/stock_oracle

/usr/bin/python3.8 scripts/update_stock_prices.py > "$LOG_FILE" 2>&1
```

Skrypt pobiera ceny akcji z biblioteki yfinance. Codziennie o 7.00 rano.

```
crontab -e

0 7 * * * /home/oracle/stock_oracle/run_cron_stock.sh >> /home/oracle/stock_oracle/cron_debug.log 2>&1
```

logi:

```
 1 Znaleziono 17 firm w bazie.
 2 Kurs USD/PLN: 3.7606
 3 Kurs USD/PLN: 3.7606
 4 Pobieram notowania dla spółki ADX (company_id=41)...
 5 Zarchiwizowano dane do: scripts/../archive/ADX_2025-04-12.csv
 6 Dodano 0 wierszy dla ADX.
 7 Pobieram notowania dla spółki AMZN (company_id=1)...
 8 Zarchiwizowano dane do: scripts/../archive/AMZN_2025-04-12.csv
 9 Dodano 0 wierszy dla AMZN.
10 Pobieram notowania dla spółki BABA (company_id=4)...
11 Zarchiwizowano dane do: scripts/../archive/BABA_2025-04-12.csv
12 Dodano 0 wierszy dla BABA.
13 Pobieram notowania dla spółki GOOGL (company_id=2)...
14 Zarchiwizowano dane do: scripts/../archive/GOOGL_2025-04-12.csv
15 Dodano 0 wierszy dla GOOGL.
16 Pobieram notowania dla spółki HSBC (company_id=13)...
17 Zarchiwizowano dane do: scripts/../archive/HSBC_2025-04-12.csv
18 Dodano 0 wierszy dla HSBC.
19 Pobieram notowania dla spółki JNJ (company_id=5)...
20 Zarchiwizowano dane do: scripts/../archive/JNJ_2025-04-12.csv
21 Dodano 0 wierszy dla JNJ.
22 Pobieram notowania dla spółki JPM (company_id=6)...
23 Zarchiwizowano dane do: scripts/../archive/JPM_2025-04-12.csv
24 Dodano 0 wierszy dla JPM.
25 Pobieram notowania dla spółki META (company_id=3)...
26 Zarchiwizowano dane do: scripts/../archive/META_2025-04-12.csv
```

	PRICE_ID	COMPANY_ID	TRADE_DATE	OPEN_PRICE	HIGH_PRICE	LOW_PRICE	CLOSE_PRICE	VOLUME	CURRENCY	CLOSE_PRICE_PLN	SOURCE
1	86	41	25/04/11	18	18,48	17,92	18,35	158900	USD	69,01	yfinance
2	91	1	25/04/11	179,93	185,86	178	184,87	50542500	USD	695,22	yfinance
3	96	4	25/04/11	104,78	108,04	102,19	107,73	40452200	USD	405,13	yfinance
4	101	2	25/04/11	152,9	157,67	152,82	157,14	33595900	USD	590,94	yfinance
5	106	13	25/04/11	48,78	50,13	48,65	49,89	3761500	USD	187,62	yfinance
6	111	5	25/04/11	149,33	152,35	148,19	151,73	9940400	USD	570,6	yfinance
7	116	6	25/04/11	226,31	238,58	225	236,2	20277100	USD	888,25	yfinance
8	121	3	25/04/11	535,51	547,4	528,59	543,57	17623700	USD	2044,15	yfinance
9	126	11	25/04/11	84,8	85,24	84,05	84,98	5001517	USD	319,58	yfinance
10	131	9	25/04/11	108,5	111,55	107,48	110,93	312853800	USD	417,16	yfinance
11	136	8	25/04/11	163,37	168,02	162,62	166,91	8602800	USD	627,68	yfinance
12	141	12	25/04/11	245,2	246,1	239,8	243,8	1333341	USD	916,83	yfinance
13	146	17	25/04/11	6,3	6,49	6,26	6,43	7852900	USD	24,18	yfinance
14	151	15	25/04/11	101,54	104,24	101,19	104	715700	USD	391,1	yfinance
15	156	10	25/04/11	167,44	170	165,86	169,42	486800	USD	637,12	yfinance
16	161	14	25/04/11	55,46	56,52	54,94	56,23	2195800	USD	211,46	yfinance
17	166	7	25/04/11	324,43	336,98	322,54	333,4	6802300	USD	1253,78	yfinance

Dodatkowo archiwizuje wczytane dane w csv w folderze archive:

Sprawdzanie danych odbywa się bezpośrednio w skrypcie:

- sprawdzanie braków Nan w kolumnach:

```
hist = hist.dropna(subset=['Open', 'High', 'Low', 'Close', 'Volume'])
```

Pomija rekordy gdzie nie ma danych.

- Sprawdzenie logicznej poprawności wartości:

```
if open_price < 0 or high_price < 0 or low_price < 0 or close_usd < 0 or
volume < 0:
    print(f"Nieprawidłowe dane dla {ticker} na {trade_date}. Pomijam.")
    continue
```

- Sprawdzenie, czy data nie jest z przyszłości:

```
if trade_date > datetime.date.today():
    print(f"Data z przyszłości ({trade_date}) — pomijam.")
    continue
```

- Sprawdzenie, czy wartości są nieujemne.

```
if volume < 0:
    print(f"Wolumen ujemny dla {ticker} - pomijam.")
    continue
```

SKRYPT AKTUALIZUJĄCY KURSY WALUT

Skrypt pobiera dane z pliku csv z NBP.

```
/usr/bin/python3.8 /home/oracle/stock_oracle/scripts/import_nbp_csv_auto.py
Plik CSV pobrany: /home/oracle/stock_oracle/scripts/../tmp/archiwum_tab_a_2025.csv
Skopiowano do archiwum: /home/oracle/stock_oracle/scripts/../archive/nbp/archiwum_tab_a_2025.csv
```

Dane są archowizowane w folderze archive/nbp

Poprawność danych jest sprawdzana,

- Filtruje puste lub niepoprawne wiersze.

```
for row in reader:
    if not row or not row[0].isdigit():
        continue
```

- Import tylko dzisiejszych kursów

```
if rate_date != datetime.date.today():
    continue   # tylko dzisiejsze kursy
```

- Sprawdzenie, czy wartość kursu istnieje i jest prawidłowa

```
rate_str = row[idx].strip().replace(",", ".")
if rate_str:
    rate = float(rate_str)
```

Dodatkowo procedura PL/SQL posiada dodatkowe zabezpieczenia przed duplikatami i rejestruje błędy w LOG. Procedury zostaną umówione w dalszej części.

PROCEDURY, FUNKCJE, WYZWALACZE OBSŁUGUJĄCE BAZĘ

- Dodawanie, usuwanie, aktualizacja rekordów

Przykładowa procedura ADD_COMPANY. Wywołanie procedury powoduje zapisanie informacji w tabeli LOG, co pozwala na analizę historii operacj, debugowanie błędów itd. Zawiera również własny wyjątek RAISE_APPLICATION_ERROR.

```
1  create or replace PROCEDURE ADD_COMPANY (
2      p_name      IN VARCHAR2,
3      p_ticker    IN VARCHAR2,
4      p_sector    IN VARCHAR2,
5      p_country   IN VARCHAR2,
6      p_website   IN VARCHAR2
7  ) AS
8      v_err_msg  VARCHAR2(4000);
9      PRAGMA AUTONOMOUS_TRANSACTION;
0  BEGIN
1      INSERT INTO Company (name, ticker, sector, country, website)
2      VALUES (p_name, p_ticker, p_sector, p_country, p_website);
3
4      -- log sukcesu
5      INSERT INTO Log (status, operation, user_name, table_name, action_detail, message)
6      VALUES ('OK','INSERT', SYS_CONTEXT('USERENV', 'SESSION_USER'), 'COMPANY', 'add_company', 'Dodano firmę: ' || p_name);
7
8      COMMIT;
9  EXCEPTION
0      WHEN OTHERS THEN
1          v_err_msg := SQLERRM;
2
3          -- log błędu
4          BEGIN
5              INSERT INTO Log (status, operation, user_name, table_name, action_detail, message)
6              VALUES ('ERROR','INSERT', SYS_CONTEXT('USERENV', 'SESSION_USER'), 'COMPANY', 'add_company', TO_CLOB(v_err_msg));
7              COMMIT;
8          EXCEPTION
9              WHEN OTHERS THEN
0                  NULL;
1          END;
2
3          RAISE_APPLICATION_ERROR(-20001, 'Błąd podczas dodawania firmy: ' || v_err_msg);
```

Przykładowy zrzut z LOG:

Pozostałe procedury i triggery:

- Dodawanie i usuwanie firmy, przenoszenie do COMPANY_ARCHIEVE po usunięciu firmy

```
1  CREATE OR REPLACE TRIGGER "TRG_ARCHIVE_COMPANY"
2  BEFORE DELETE ON COMPANY
3  FOR EACH ROW
4  DECLARE
5      v_user VARCHAR2(50);
6  BEGIN
7      v_user := SYS_CONTEXT('USERENV', 'SESSION_USER');
8      INSERT INTO COMPANY_ARCHIVE
9         (COMPANY_ID, NAME, TICKER, SECTOR, COUNTRY, WEBSITE, DELETED_AT, DELETED_BY)
10     VALUES
11        (:OLD.COMPANY_ID, :OLD.NAME, :OLD.TICKER, :OLD.SECTOR, :OLD.COUNTRY, :OLD.WEBSITE, SYSTIMESTAMP, v_user);
12 END;
13
14
```

TEST:

```
Worksheet    Query Builder

 1 □ BEGIN
 2     ADD_COMPANY(
 3       p_name    => 'FIRMA',
 4       p_ticker  => 'FIR',
 5       p_sector  => 'NAUKA',
 6       p_country => 'PL',
 7       p_website => 'https://www.firma.com'
 8     );
 9   END;
10
```

Script Output ×

📌 🧽 💾 🖨 📑 | Task completed in 0,08 seconds

PL/SQL procedure successfully completed.

```
12 □ BEGIN
13     delete_company(
14       p_id    => 61,
15       p_name  => 'FIRMA'
16     );
17   END;
18
```

COMPANY_ARCHIVE

	61 FIRMA	FIRMA FIRMA		FIRMA FIRMA		25/04/05 08.15.40,410014000 KAROL	
18	61 FIRMA	FIR	NAUKA	PL	https://www.firma.com 25/04/12 10:25:10,969714000 KAROL	

LOG

	LOG_ID	LOG_TIME	STATUS	OPERATION	USER_NAME	TABLE_NAME	ACTION_DETAIL	MESSAGE
1	665	25/04/12 10:25:10,980114000	OK	DELETE	KAROL	COMPANY	delete_company	Usunięto firmę: FIRMA
2	664	25/04/12 10:25:10,967354000	OK	DELETE	KAROL	COMPANY	Usunięto rekord COMPANY_ID=61 Rekord przeniesiony do COMPANY_ARCHIVE	

Podobnie działają procedury EDIT_COMPANY, ADD_INVESTOR, DELETE_INVESTOR, EDIT_INVESTOR

Walidacja inwestora:

```sql
create or replace TRIGGER trg_validate_investor
BEFORE INSERT OR UPDATE ON Investor
FOR EACH ROW
DECLARE
    invalid_email EXCEPTION;
    invalid_phone EXCEPTION;
    invalid_nid   EXCEPTION;
BEGIN
    -- Walidacja e-maila
    IF NOT REGEXP_LIKE(:NEW.email, '^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,}$') THEN
        RAISE invalid_email;
    END IF;

    -- Walidacja numeru telefonu (9 cyfr lub zaczynający się od +XX)
    IF :NEW.phone IS NOT NULL AND NOT REGEXP_LIKE(:NEW.phone, '^\+?\d{9,15}$') THEN
        RAISE invalid_phone;
    END IF;

    -- Walidacja PESEL (11 cyfr + kontrola długości)
    IF :NEW.national_id IS NOT NULL AND NOT REGEXP_LIKE(:NEW.national_id, '^\d{11}$') THEN
        RAISE invalid_nid;
    END IF;

EXCEPTION
    WHEN invalid_email THEN
        RAISE_APPLICATION_ERROR(-20001, 'Niepoprawny adres e-mail');
    WHEN invalid_phone THEN
        RAISE_APPLICATION_ERROR(-20002, 'Niepoprawny numer telefonu');
    WHEN invalid_nid THEN
        RAISE_APPLICATION_ERROR(-20003, 'Niepoprawny numer identyfikacyjny (PESEL)');
END;
```

```sql
34 BEGIN
35   ADD_INVESTOR(
36     p_client_code => 'CL9999',
37     p_name        => 'Jan Kowalski',
38     p_email       => 'jan.kowalski@sql.com',
39     p_phone       => '+481125',
40     p_national_id => '85010112345'
41   );
42 END;
43 /
44
45
46
47
```

Script Output × | Query Result ×

📌 ✏ 💾 🖨 📋 | Task completed in 0,057 seconds

```
ERROR at line 1:
ORA-20001: Błąd podczas dodawania inwestora: ORA-20002: Niepoprawny numer telefonu
ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 25
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
ORA-06512: przy "KAROL.ADD_INVESTOR", linia 33
ORA-06512: przy linia 2
```

```
34 BEGIN
35   ADD_INVESTOR(
36     p_client_code => 'CL9999',
37     p_name        => 'Jan Kowalski',
38     p_email       => 'jan.kowalski@sql.com',
39     p_phone       => '+48123456789',
40     p_national_id => '123'
41   );
42 END;
43 /
44
45
46
47
```

Script Output × | Query Result ×

Task completed in 0,062 seconds

```
ERROR at line 1:
ORA-20001: Błąd podczas dodawania inwestora: ORA-20003: Niepoprawny numer identyfikacyjny (PESEL)
ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 27
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
ORA-06512: przy "KAROL.ADD_INVESTOR", linia 33
ORA-06512: przy linia 2
```

```
34 BEGIN
35   ADD_INVESTOR(
36     p_client_code => 'CL9999',
37     p_name        => 'Jan Kowalski',
38     p_email       => 'jan.kowalski-sql.com',
39     p_phone       => '+48123456789',
40     p_national_id => '85010112345'
41   );
42 END;
43 /
44
45
46
47
```

Script Output × | Query Result ×

Task completed in 0,048 seconds

```
ERROR at line 1:
ORA-20001: Błąd podczas dodawania inwestora: ORA-20001: Niepoprawny adres e-mail
ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 23
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
ORA-06512: przy "KAROL.ADD_INVESTOR", linia 33
ORA-06512: przy linia 2
```

```
34  BEGIN
35    ADD_INVESTOR(
36      p_client_code => 'CL9999',
37      p_name        => 'Jan Kowalski',
38      p_email       => 'jan.kowalski@sql.com',
39      p_phone       => '+48123456789',
40      p_national_id => '85010112345'
41    );
42  END;
43  /
44
45
46
47
```

Script Output ×   Query Result ×

Task completed in 0,057 seconds

```
ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 23
ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
ORA-06512: przy "KAROL.ADD_INVESTOR", linia 33
ORA-06512: przy linia 2

https://docs.oracle.com/error-help/db/ora-20001/


More Details :
https://docs.oracle.com/error-help/db/ora-20001/
https://docs.oracle.com/error-help/db/ora-06512/
https://docs.oracle.com/error-help/db/ora-04088/


PL/SQL procedure successfully completed.
```

- Procedura MAKE_TRANSACTION

```sql
create or replace PROCEDURE make_transaction (
    p_investor_id      IN NUMBER,
    p_company_id       IN NUMBER,
    p_operation        IN VARCHAR2,  -- 'BUY' lub 'SELL'
    p_shares           IN NUMBER,
    p_price_per_share  IN NUMBER,
    p_commission_pct   IN NUMBER DEFAULT 0
) AS
    v_total_value    NUMBER(12, 2);
    v_err_msg        VARCHAR2(4000);
    v_existing_shares NUMBER;
    PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
    -- Oblicz wartość transakcji z uwzględnieniem prowizji
    v_total_value := p_shares * p_price_per_share * (1 + p_commission_pct / 100);

    IF p_operation = 'SELL' THEN
        -- Sprawdź czy inwestor ma wystarczającą liczbę akcji
        SELECT NVL(shares, 0) INTO v_existing_shares
        FROM Portfolio
        WHERE investor_id = p_investor_id AND company_id = p_company_id;

        IF v_existing_shares < p_shares THEN
            RAISE_APPLICATION_ERROR(-20005, 'Nie można sprzedać więcej akcji niż posiadanych.');
        END IF;

        -- Zmniejsz stan posiadania
        UPDATE Portfolio
        SET shares = shares - p_shares,
            avg_price = CASE WHEN shares - p_shares > 0 THEN avg_price ELSE NULL END,
            last_updated = CURRENT_TIMESTAMP
        WHERE investor_id = p_investor_id AND company_id = p_company_id;
```

```sql
    ELSIF p_operation = 'BUY' THEN
        -- Jeśli istnieje rekord, aktualizuj; jeśli nie, dodaj
        BEGIN
            SELECT shares INTO v_existing_shares
            FROM Portfolio
            WHERE investor_id = p_investor_id AND company_id = p_company_id;

            UPDATE Portfolio
            SET
                shares = shares + p_shares,
                avg_price = ROUND(((avg_price * shares) + (p_price_per_share * p_shares)) / (shares +
p_shares), 2),
                last_updated = CURRENT_TIMESTAMP
            WHERE investor_id = p_investor_id AND company_id = p_company_id;

        EXCEPTION
            WHEN NO_DATA_FOUND THEN
                INSERT INTO Portfolio (investor_id, company_id, shares, avg_price)
                VALUES (p_investor_id, p_company_id, p_shares, p_price_per_share);
        END;
    ELSE
        RAISE_APPLICATION_ERROR(-20006, 'Nieznana operacja. Dozwolone: BUY lub SELL.');
    END IF;

    -- Dodaj wpis transakcji
    INSERT INTO Transaction (
        investor_id, company_id, operation, shares,
        price_per_share, total_value, transaction_date
    ) VALUES (
        p_investor_id, p_company_id, p_operation, p_shares,
        p_price_per_share, v_total_value, SYSDATE
    );

    -- Dodaj log
    INSERT INTO Log (status, operation, user_name, table_name, action_detail, message)
    VALUES ('OK', 'INSERT', SYS_CONTEXT('USERENV', 'SESSION_USER'), 'TRANSACTION',
            'make_transaction', 'Transakcja: ' || p_operation || ' ' || p_shares || ' akcji');

    COMMIT;

EXCEPTION
    WHEN OTHERS THEN
        v_err_msg := SQLERRM;
        BEGIN
            INSERT INTO Log (status, operation, user_name, table_name, action_detail, message)
            VALUES ('ERROR', 'INSERT', SYS_CONTEXT('USERENV', 'SESSION_USER'), 'TRANSACTION',
                    'make_transaction', TO_CLOB(v_err_msg));
            COMMIT;
        EXCEPTION
            WHEN OTHERS THEN NULL;
        END;
        RAISE_APPLICATION_ERROR(-20010, 'Błąd transakcji: ' || v_err_msg);
END;
```

```sql
1 BEGIN
2     make_transaction(
3         p_investor_id      => 1,
4         p_company_id       => 5,
5         p_operation        => 'BUY',
6         p_shares           => 100,
7         p_price_per_share  => 150.25,
8         p_commission_pct   => 0.5
9     );
0 END;
1 /
```

TRANSACTION_ID	INVESTOR_ID	COMPANY_ID	OPERATION	SHARES	PRICE_PER_SHARE	COMMISSION_RATE	TOTAL_VALUE	TRANSACTION_DATE
2	1	5	BUY	100	150,25	0	15100,13	25/04/12

PORTFOLIO_ID	INVESTOR_ID	COMPANY_ID	SHARES	AVG_PRICE	LAST_UPDATED
21	1	5	200	150,25	25/04/12 10:38:10,761098000
3	1	15	5	108,29	25/04/04 11:54:16.557623000

- Procedura GENERATE_DAILY_PORTFOLIO_SNAPSHOTS

```sql
1  create or replace PROCEDURE generate_daily_portfolio_snapshots IS
2     CURSOR c_investors IS
3        SELECT investor_id FROM INVESTOR;
4
5     v_total_value NUMBER(12, 2);
6  BEGIN
7     FOR inv IN c_investors LOOP
8        SELECT SUM(p.shares * get_current_stock_value(p.company_id))
9        INTO v_total_value
10       FROM PORTFOLIO p
11       WHERE p.investor_id = inv.investor_id;
12
13       INSERT INTO PORTFOLIOSNAPSHOT (INVESTOR_ID, SNAPSHOT_DATE, TOTAL_VALUE_PLN)
14       VALUES (inv.investor_id, SYSDATE, v_total_value);
15    END LOOP;
16 END;
17
```

DO wywołania jej wykorzystujemy DBMS_SCHEDULER

```sql
BEGIN
  DBMS_SCHEDULER.CREATE_JOB (
    job_name         => 'JOB_SNAPSHOT_TEST',
    job_type         => 'STORED_PROCEDURE',
    job_action       => 'generate_daily_portfolio_snapshots',
    start_date       => SYSTIMESTAMP,
    repeat_interval  => 'FREQ=DAILY; BYHOUR=23; BYMINUTE=59',
    enabled          => TRUE
  );
END;
/
```

```sql
17  SELECT JOB_NAME, ENABLED, STATE, LAST_START_DATE, NEXT_RUN_DATE
18  FROM USER_SCHEDULER_JOBS;
19
```

Script Output × ► Query Result ×

📌 🖨 🗄 📋 SQL | All Rows Fetched: 3 in 0,178 seconds

	JOB_NAME	ENABLED	STATE	LAST_START_DATE	NEXT_RUN_DATE
1	JOB_SNAPSHOT_TEST	TRUE	SCHEDULED	25/04/11 23:59:37,156158000 +02:00	25/04/12 23:59:37,158005000 +02:00
2	JOB_TEST_INSERT	TRUE	SCHEDULED	25/04/12 10:44:41,246228000 +02:00	25/04/12 10:45:41,262181000 +02:00
3	JOB_TEST_MINUTELY	TRUE	SCHEDULED	25/04/12 10:44:53,335410000 +02:00	25/04/12 10:45:53,337014000 +02:00

Natychmiastowe wywołanie:

```sql
BEGIN
  DBMS_SCHEDULER.RUN_JOB('JOB_SNAPSHOT_TEST');
END;
/
```

```
25  SELECT JOB_NAME, STATUS, ACTUAL_START_DATE, RUN_DURATION, ERROR#
26  FROM USER_SCHEDULER_JOB_RUN_DETAILS
27  ORDER BY ACTUAL_START_DATE DESC;
28
29
```

Script Output ×   Query Result ×

SQL | Fetched 50 rows in 0,019 seconds

	JOB_NAME	STATUS	ACTUAL_START_DATE	RUN_DURATION	ERROR#
1	JOB_TEST_MINUTELY	SUCCEEDED	25/04/12 10:49:53,505401000 +02:00	+00 00:00:00.000000	0
2	JOB_SNAPSHOT_TEST	SUCCEEDED	25/04/12 10:49:47,462997000 +02:00	+00 00:00:00.000000	0

```
CREATE OR REPLACE TRIGGER trg_check_snapshot_type
BEFORE INSERT OR UPDATE ON PORTFOLIOSNAPSHOT
FOR EACH ROW
BEGIN
    IF :NEW.SNAPSHOT_TYPE IS NULL OR UPPER(:NEW.SNAPSHOT_TYPE) NOT IN ('DAILY', 'MANUAL', 'SYSTEM') THEN
        RAISE_APPLICATION_ERROR(-20001, 'Nieprawidłowy typ SNAPSHOT_TYPE. Dozwolone: DAILY, MANUAL, SYSTEM.');
    END IF;
END;
```

	SNAPSHOT_ID	INVESTOR_ID	SNAPSHOT_...	TOTAL_VALUE_PLN	SNAPSHOT_TYPE
1	116	4	25/04/12	7263,6	DAILY
2	117	5	25/04/12	22938	DAILY
3	114	2	25/04/12	59460,8	DAILY
4	118	8	25/04/12	(null)	DAILY
5	113	1	25/04/12	134770,75	DAILY
6	115	3	25/04/12	(null)	DAILY

- Procedura GENERATE_PORTFOLIO_SUMMARY

```
create or replace PROCEDURE generate_portfolio_summary(
    p_period_type IN VARCHAR2,
    p_investor_id IN NUMBER DEFAULT NULL
) IS
BEGIN
  MERGE INTO PORTFOLIO_SUMMARY s
  USING (
    SELECT
      investor_id,
      UPPER(p_period_type) AS period_type,
      MAX(snapshot_date) AS latest_date,
      TO_NUMBER(TO_CHAR(MAX(snapshot_date), 'YYYY')) AS period_year,
      CASE UPPER(p_period_type)
        WHEN 'MONTH' THEN TO_NUMBER(TO_CHAR(MAX(snapshot_date), 'MM'))
        WHEN 'QUARTER' THEN CEIL(TO_NUMBER(TO_CHAR(MAX(snapshot_date), 'MM')) / 3)
        WHEN 'YEAR' THEN 1
      END AS period_number,
      CASE UPPER(p_period_type)
        WHEN 'MONTH' THEN TO_CHAR(MAX(snapshot_date), 'YYYY-MM')
        WHEN 'QUARTER' THEN TO_CHAR(MAX(snapshot_date), 'YYYY') || '-Q' ||
TO_CHAR(CEIL(TO_NUMBER(TO_CHAR(MAX(snapshot_date), 'MM')) / 3))
        WHEN 'YEAR' THEN TO_CHAR(MAX(snapshot_date), 'YYYY')
      END AS period_label
    FROM (
      SELECT investor_id, snapshot_date, total_value_pln FROM PORTFOLIOSNAPSHOT
      WHERE p_investor_id IS NULL OR investor_id = p_investor_id
      UNION ALL
      SELECT investor_id, snapshot_date, total_value_pln FROM PORTFOLIOSNAPSHOT_ARCHIVE
      WHERE p_investor_id IS NULL OR investor_id = p_investor_id
    )
    GROUP BY investor_id,
      CASE UPPER(p_period_type)
        WHEN 'MONTH' THEN TO_CHAR(snapshot_date, 'YYYY-MM')
```

```sql
        WHEN 'QUARTER' THEN TO_CHAR(snapshot_date, 'YYYY') || '-Q' ||
TO_CHAR(CEIL(TO_NUMBER(TO_CHAR(snapshot_date, 'MM')) / 3))
        WHEN 'YEAR' THEN TO_CHAR(snapshot_date, 'YYYY')
      END
  ) latest
  ON (
    s.investor_id = latest.investor_id AND
    s.period_type = latest.period_type AND
    s.period_label = latest.period_label
  )
  WHEN MATCHED THEN
    UPDATE SET
      total_value_pln = (
        SELECT total_value_pln FROM (
          SELECT investor_id, snapshot_date, total_value_pln FROM PORTFOLIOSNAPSHOT
          WHERE p_investor_id IS NULL OR investor_id = latest.investor_id
          UNION ALL
          SELECT investor_id, snapshot_date, total_value_pln FROM PORTFOLIOSNAPSHOT_ARCHIVE
          WHERE p_investor_id IS NULL OR investor_id = latest.investor_id
        )
        WHERE investor_id = latest.investor_id AND snapshot_date = latest.latest_date
        FETCH FIRST 1 ROWS ONLY
      ),
      average_price_pln = get_average_value_pln(
        latest.investor_id,
        latest.period_type,
        latest.period_label
      ),
      generated_at = SYSTIMESTAMP
  WHEN NOT MATCHED THEN
    INSERT (
      investor_id,
      period_type,
      period_label,
      period_year,
      period_number,
      total_value_pln,
      average_price_pln,
      generated_at
    ) VALUES (
      latest.investor_id,
      latest.period_type,
      latest.period_label,
      latest.period_year,
      latest.period_number,
      (
        SELECT total_value_pln FROM (
          SELECT investor_id, snapshot_date, total_value_pln FROM PORTFOLIOSNAPSHOT
          WHERE p_investor_id IS NULL OR investor_id = latest.investor_id
          UNION ALL
          SELECT investor_id, snapshot_date, total_value_pln FROM PORTFOLIOSNAPSHOT_ARCHIVE
          WHERE p_investor_id IS NULL OR investor_id = latest.investor_id
        )
        WHERE investor_id = latest.investor_id AND snapshot_date = latest.latest_date
        FETCH FIRST 1 ROWS ONLY
      ),
      get_average_value_pln(
        latest.investor_id,
        latest.period_type,
        latest.period_label
      ),
      SYSTIMESTAMP
    );
END;
```

Dla wywołania:

```
BEGIN
  generate_portfolio_summary('MONTH');
END;
/
```

Zwróci w tabeli PORTFOLIO_SUMMARY

	SUMMARY_ID	INVESTOR_ID	PERIOD_TYPE	PERIOD_YEAR	PERIOD_NUMBER	PERIOD_LABEL	TOTAL_VALUE_PLN	GENERATED_AT	AVERAGE_PRICE_PLN
1	30	3 MONTH	2025	2	2025-02	4594	25/04/12 11:36:16,843445000	7178,43
2	51	1 MONTH	2025	2	2025-02	5759	25/04/12 11:36:16,843445000	7395,71
3	50	1 MONTH	2025	1	2025-01	5257	25/04/12 11:36:16,843445000	7121
4	49	5 MONTH	2025	1	2025-01	5777	25/04/12 11:36:16,843445000	7627,6
5	48	4 MONTH	2024	11	2024-11	6503	25/04/12 11:36:16,843445000	7076,2
6	47	3 MONTH	2025	1	2025-01	5884	25/04/12 11:36:16,843445000	6223,8
7	46	4 MONTH	2025	2	2025-02	6185	25/04/12 11:36:16,843445000	6699,86
8	45	5 MONTH	2025	2	2025-02	5895	25/04/12 11:36:16,843445000	7615
9	44	5 MONTH	2024	12	2024-12	4318	25/04/12 11:36:16,843445000	6836,33
10	43	4 MONTH	2024	12	2024-12	6626	25/04/12 11:36:16,843445000	5612,83
11	42	3 MONTH	2025	3	2025-03	9379	25/04/12 11:36:16,843445000	6769
12	41	5 MONTH	2025	3	2025-03	5107	25/04/12 11:36:16,843445000	6762,22
13	40	3 MONTH	2024	11	2024-11	5773	25/04/12 11:36:16,843445000	7480,4
14	39	22 MONTH	2024	11	2024-11	4649	25/04/12 11:36:16,843445000	4649
15	38	2 MONTH	2025	2	2025-02	51000	25/04/12 11:36:16,843445000	51077,29
16	37	2 MONTH	2024	11	2024-11	39878	25/04/12 11:36:16,843445000	39647
17	36	1 MONTH	2024	12	2024-12	8683	25/04/12 11:36:16,843445000	6456,33
18	35	4 MONTH	2025	3	2025-03	7120	25/04/12 11:36:16,843445000	7218,11
19	27	2 MONTH	2025	3	2025-03	32354	25/04/12 11:36:16,843445000	43802,11
20	28	5 MONTH	2024	11	2024-11	8412	25/04/12 11:36:16,843445000	8453
21	29	3 MONTH	2024	12	2024-12	7365	25/04/12 11:36:16,843445000	7077,83
22	52	1 MONTH	2025	3	2025-03	4279	25/04/12 11:36:16,843445000	7783,44
23	31	4 MONTH	2025	1	2025-01	8844	25/04/12 11:36:16,843445000	7063,2
24	32	1 MONTH	2024	11	2024-11	7510	25/04/12 11:36:16,843445000	6741,25
25	33	2 MONTH	2024	12	2024-12	41258	25/04/12 11:36:16,843445000	38679,83
26	34	2 MONTH	2025	1	2025-01	49008	25/04/12 11:36:16,843445000	47003,8

```
BEGIN
  generate_portfolio_summary('YEAR');
END;
/
```

	SUMMARY_ID	INVESTOR_ID	PERIOD_TYPE	PERIOD_YEAR	PERIOD_NUMBER	PERIOD_LABEL	TOTAL_VALUE_PLN	GENERATED_AT	AVERAGE_PRICE_PLN
27	17	8 YEAR	2025	1	2025	(null)	25/04/11 18:32:52,774481000	(null)
28	18	3 YEAR	2025	1	2025	(null)	25/04/11 18:32:52,774481000	6212
29	16	4 YEAR	2025	1	2025	6881,5	25/04/11 18:32:52,774481000	7202,63
30	15	2 YEAR	2025	1	2025	51888,5	25/04/11 18:32:52,774481000	48970,13
31	14	1 YEAR	2025	1	2025	5788,42	25/04/11 18:32:52,774481000	5788,42
32	13	5 YEAR	2025	1	2025	20292,8	25/04/11 18:32:52,774481000	17540,6

```
BEGIN
  generate_portfolio_summary('QUARTER');
END;
/
```

	SUMMAR...	INVESTOR_ID	PERIOD_TYPE	PERIOD_YEAR	PERIOD_NUMBER	PERIOD_LABEL	TOTAL_VALUE_PLN	GENERATED_AT	AVERAGE_PRICE_PLN
1	81	4 QUARTER	2025	1	2025-Q1	7120	25/04/12 11:40:19,901830000	7008,48
2	80	4 QUARTER	2024	4	2024-Q4	6626	25/04/12 11:40:19,901830000	6278
3	79	1 QUARTER	2024	4	2024-Q4	8683	25/04/12 11:40:19,901830000	6570,3
4	78	22 QUARTER	2024	4	2024-Q4	4649	25/04/12 11:40:19,901830000	4649
5	77	5 QUARTER	2025	1	2025-Q1	5107	25/04/12 11:40:19,901830000	7252,52
6	76	3 QUARTER	2024	4	2024-Q4	7365	25/04/12 11:40:19,901830000	7260,82
7	75	5 QUARTER	2024	4	2024-Q4	4318	25/04/12 11:40:19,901830000	7571,18
8	74	1 QUARTER	2025	1	2025-Q1	4279	25/04/12 11:40:19,901830000	7496,48
9	73	3 QUARTER	2025	1	2025-Q1	9379	25/04/12 11:40:19,901830000	6775,67
10	72	2 QUARTER	2024	4	2024-Q4	41258	25/04/12 11:40:19,901830000	39119,45
11	71	2 QUARTER	2025	1	2025-Q1	32354	25/04/12 11:40:19,901830000	46989,48

Procedurę można też wywołać z opcjonalnym parametrem id inwestora:

```
BEGIN
   generate_portfolio_summary('MONTH', 1);
END;
/
```

Dodaliśmy trigger, aby wywoływał procedurę przy każdym dodanym nowym snapshocie.

```
CREATE OR REPLACE TRIGGER trg_generate_summary_after_snapshot
AFTER INSERT ON PORTFOLIOSNAPSHOT
FOR EACH ROW
BEGIN
   generate_portfolio_summary('MONTH', :NEW.investor_id);
   generate_portfolio_summary('QUARTER', :NEW.investor_id);
   generate_portfolio_summary('YEAR', :NEW.investor_id);
END;
/
```

- Pozostałe funckje i triggery.

```
create or replace FUNCTION calculate_new_avg_price (
    p_current_avg     IN NUMBER,
    p_current_shares  IN NUMBER,
    p_new_price       IN NUMBER,
    p_new_shares      IN NUMBER
) RETURN NUMBER
IS
    v_total_cost    NUMBER;
    v_total_shares  NUMBER;
BEGIN
    v_total_cost := (p_current_avg * p_current_shares) + (p_new_price * p_new_shares);
    v_total_shares := p_current_shares + p_new_shares;

    IF v_total_shares = 0 THEN
        RETURN 0;  -- lub NULL, jeśli nie ma akcji
    ELSE
        RETURN ROUND(v_total_cost / v_total_shares, 2);
    END IF;
END;
```

```sql
create or replace FUNCTION get_average_value_pln (
    p_investor_id    IN NUMBER,
    p_period_type    IN VARCHAR2,
    p_period_label   IN VARCHAR2
) RETURN NUMBER IS
    v_avg NUMBER(12,2);
BEGIN
    SELECT ROUND(AVG(total_value_pln), 2)
    INTO v_avg
    FROM (
        SELECT investor_id, snapshot_date, total_value_pln FROM PORTFOLIOSNAPSHOT
        UNION ALL
        SELECT investor_id, snapshot_date, total_value_pln FROM PORTFOLIOSNAPSHOT_ARCHIVE
    )
    WHERE investor_id = p_investor_id
      AND (
        (p_period_type = 'MONTH'   AND TO_CHAR(snapshot_date, 'YYYY-MM') = p_period_label) OR
        (p_period_type = 'QUARTER' AND TO_CHAR(snapshot_date, 'YYYY') || '-Q' || TO_CHAR(CEIL(TO_NUMBER(TO_CHAR(snapshot_date, 'MM')) / 3)) = p_period_label) OR
        (p_period_type = 'YEAR'    AND TO_CHAR(snapshot_date, 'YYYY') = p_period_label)
      );

    RETURN v_avg;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN NULL;
END;
```

```sql
create or replace FUNCTION get_current_stock_value(p_company_id IN NUMBER)
RETURN NUMBER
IS
    v_close_price STOCKPRICE.close_price%TYPE;
    v_rate_to_pln EXCHANGERATE.rate_to_pln%TYPE;
    v_result NUMBER;
BEGIN
    -- pobierz ostatnią cenę akcji
    SELECT close_price INTO v_close_price
    FROM STOCKPRICE
    WHERE company_id = p_company_id
    ORDER BY trade_date DESC
    FETCH FIRST 1 ROWS ONLY;

    -- pobierz kurs USD/PLN
    SELECT rate_to_pln INTO v_rate_to_pln
    FROM EXCHANGERATE
    WHERE currency = 'USD'
    ORDER BY rate_date DESC
    FETCH FIRST 1 ROWS ONLY;

    -- oblicz wartość
    v_result := ROUND(v_close_price * v_rate_to_pln, 2);

    RETURN v_result;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN NULL;
    WHEN OTHERS THEN
        RETURN NULL;
END;
```

```sql
create or replace FUNCTION get_latest_trade_date(
    p_company_id IN NUMBER
)
RETURN DATE
AS
    v_date DATE;
BEGIN
    SELECT trade_date
      INTO v_date
      FROM stockprice
     WHERE company_id = p_company_id
    ;
    RETURN v_date;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN NULL;
END;
```

```sql
create or replace TRIGGER trg_check_snapshot_type
BEFORE INSERT OR UPDATE ON PORTFOLIOSNAPSHOT
FOR EACH ROW
BEGIN
  IF :NEW.SNAPSHOT_TYPE IS NULL OR UPPER(:NEW.SNAPSHOT_TYPE) NOT IN ('DAILY', 'MANUAL', 'SYSTEM') THEN
    RAISE_APPLICATION_ERROR(-20001, 'Nieprawidłowy typ SNAPSHOT_TYPE. Dozwolone: DAILY, MANUAL, SYSTEM.');
  END IF;
END;
```

```sql
create or replace TRIGGER "TRG_COMPANY_ARCHIVE_DELETE"
BEFORE DELETE ON company
FOR EACH ROW
BEGIN
    INSERT INTO company_archive (company_id, name, ticker, sector, country, website, deleted_at, deleted_by)
    VALUES (
        :OLD.company_id,
        :OLD.name,
        :OLD.ticker,
        :OLD.sector,
        :OLD.country,
        :OLD.website,
        SYSTIMESTAMP,
        SYS_CONTEXT('USERENV','SESSION_USER')
    );

    log_company_delete(
        p_company_id    => :OLD.company_id,
        p_status        => 'OK',
        p_operation     => 'DELETE',
        p_table_name    => 'COMPANY',
        p_action_detail => 'Usunięto rekord COMPANY_ID=' || TO_CHAR(:OLD.company_id),
        p_message       => 'Rekord przeniesiony do COMPANY_ARCHIVE'
    );
EXCEPTION
    WHEN OTHERS THEN
        RAISE;
END;
```

- Widoki i funckje okienkowe.

```sql
CREATE OR REPLACE VIEW V_INVESTOR_RANK AS
SELECT
    investor_id,
    total_value_pln,
    RANK() OVER (ORDER BY total_value_pln DESC) AS rank
FROM (
    SELECT investor_id, total_value_pln
    FROM portfoliosnapshot
    WHERE snapshot_date = (
        SELECT MAX(snapshot_date) FROM portfoliosnapshot
    )
    AND total_value_pln IS NOT NULL
);
```

```
103
104  SELECT * FROM V_INVESTOR_RANK;
105
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 4 in 0,008 seconds

	INVESTOR_ID	TOTAL_VALUE_PLN	RANK
1	1	134770,75	1
2	2	59460,8	2
3	5	22938	3
4	4	7263,6	4

```sql
CREATE OR REPLACE VIEW V_PORTFOLIO_CUMULATIVE AS
SELECT investor_id, snapshot_date, total_value_pln, cumulative_value
FROM (
  SELECT
    investor_id,
    snapshot_date,
    total_value_pln,
    SUM(total_value_pln) OVER (
      PARTITION BY investor_id
      ORDER BY snapshot_date
      ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW
    ) AS cumulative_value,
    ROW_NUMBER() OVER (
      PARTITION BY investor_id, snapshot_date
      ORDER BY total_value_pln DESC
    ) AS rn
  FROM portfoliosnapshot
  WHERE total_value_pln IS NOT NULL
)
WHERE rn = 1
ORDER BY investor_id, snapshot_date;
```

```
130  SELECT * FROM V_PORTFOLIO_CUMULATIVE;
```

Script Output × Query Result ×

SQL | All Rows Fetched: 20 in 0,008 seconds

	INVESTOR_ID	SNAPSHOT_DATE	TOTAL_VALUE_PLN	CUMULATIVE_VALUE
1	1	25/04/05	5788,42	5788,42
2	1	25/04/11	5788,42	11576,84
3	1	25/04/12	134770,75	146347,59
4	1	25/04/12	134770,75	281118,34
5	2	25/04/05	51888,5	51888,5
6	2	25/04/06	40215	92103,5
7	2	25/04/11	51888,5	143992
8	2	25/04/12	59460,8	203452,8
9	2	25/04/12	59460,8	262913,6
10	3	25/04/06	6212	6212
11	4	25/04/05	6881,5	6881,5
12	4	25/04/06	8166	15047,5
13	4	25/04/11	6881,5	21929
14	4	25/04/12	7263,6	29192,6

```sql
CREATE OR REPLACE VIEW V_PORTFOLIO_DAILY_CHANGE AS
SELECT
  investor_id,
  snapshot_date,
  total_value_pln,
  LAG(total_value_pln) OVER (
    PARTITION BY investor_id
    ORDER BY snapshot_date
  ) AS previous_value,
  total_value_pln - LAG(total_value_pln) OVER (
    PARTITION BY investor_id
    ORDER BY snapshot_date
  ) AS daily_change
FROM portfoliosnapshot
WHERE total_value_pln IS NOT NULL;
```

```sql
148    SELECT * FROM V_PORTFOLIO_DAILY_CHANGE;
```

Script Output ✕   Query Result ✕

SQL | All Rows Fetched: 20 in 0,009 seconds

	INVESTOR_ID	SNAPSHOT_DATE	TOTAL_VALUE_PLN	PREVIOUS_VALUE	DAILY_CHANGE
1	1	25/04/05	5788,42	(null)	(null)
2	1	25/04/11	5788,42	5788,42	0
3	1	25/04/12	134770,75	5788,42	128982,33
4	1	25/04/12	134770,75	134770,75	0
5	2	25/04/05	51888,5	(null)	(null)
6	2	25/04/06	40215	51888,5	-11673,5
7	2	25/04/11	51888,5	40215	11673,5
8	2	25/04/12	59460,8	51888,5	7572,3
9	2	25/04/12	59460,8	59460,8	0
10	3	25/04/06	6212	(null)	(null)
11	4	25/04/05	6881,5	(null)	(null)
12	4	25/04/06	8166	6881,5	1284,5
13	4	25/04/11	6881,5	8166	-1284,5
14	4	25/04/12	7263,6	6881,5	382,1
15	4	25/04/12	7263,6	7263,6	0
16	5	25/04/05	20292,8	(null)	(null)

```sql
CREATE OR REPLACE VIEW v_current_portfolio_value AS
SELECT
    p.investor_id,
```

```sql
        i.name AS investor_name,
        i.email AS investor_email,
        i.client_code,
        SUM(
            p.shares
            * (SELECT s.close_price
                FROM STOCKPRICE s
                WHERE s.company_id = p.company_id
                ORDER BY s.trade_date DESC
                FETCH FIRST 1 ROWS ONLY)
            * (SELECT e.rate_to_pln
                FROM EXCHANGERATE e
                WHERE e.currency = 'USD'
                ORDER BY e.rate_date DESC
                FETCH FIRST 1 ROWS ONLY)
        ) AS total_value_pln
FROM PORTFOLIO p
JOIN INVESTOR i ON p.investor_id = i.investor_id
GROUP BY p.investor_id, i.name, i.email, i.client_code;
```

```
150  SELECT * FROM v_current_portfolio_value;
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 4 in 0,029 seconds

	INVESTOR_ID	INVESTOR_NAME	INVESTOR_EMAIL	CLIENT_CODE	TOTAL_VALUE_PLN
1	1	Jan Kowalski	jan.kowalski@f1.com	CL111	134771,610672
2	2	Anna Nowak	anna.nowak@sql.com	CL222	59460,82038
3	4	Katarzyna Zielińska	katarzyna.zielinska@sql.com	CL004	7263,45066
4	5	Tomasz Wójcik	tomasz.wojcik@sql.com	CL005	22937,94594

```sql
1  CREATE OR REPLACE FUNCTION get_exchange_rate(
2      p_currency IN VARCHAR2
3  ) RETURN NUMBER
4  IS
5      v_rate EXCHANGERATE.RATE_TO_PLN%TYPE;
6  BEGIN
7      SELECT RATE_TO_PLN
8      INTO v_rate
9      FROM EXCHANGERATE
0      WHERE CURRENCY = UPPER(p_currency)
1      ORDER BY RATE_DATE DESC
2      FETCH FIRST 1 ROWS ONLY;
3
4      RETURN v_rate;
5  EXCEPTION
6      WHEN NO_DATA_FOUND THEN
7          RETURN NULL;
8      WHEN OTHERS THEN
9          RETURN NULL;
0  END;
1  /
2
3  SELECT get_exchange_rate('USD') FROM dual;
4
```

Script Output × | Query Result ×

SQL | All Rows Fetched: 1 in 0,028 seconds

	GET_EXCHANGE_RATE('USD')
1	4,2222

```
Worksheet    Query Builder
 1 □CREATE OR REPLACE VIEW v_company_stock_prices_all AS
 2  SELECT
 3      c.company_id,
 4      c.name AS company_name,
 5      c.ticker,
 6      s.trade_date,
 7      s.open_price,
 8      s.high_price,
 9      s.low_price,
10      s.close_price,
11      ROUND(s.close_price * get_exchange_rate('USD'), 2) AS close_price_pln,
12      s.volume
13  FROM (
14      SELECT company_id, trade_date, open_price, high_price, low_price, close_price, volume
15      FROM STOCKPRICE
16      UNION ALL
17      SELECT company_id, trade_date, open_price, high_price, low_price, close_price, volume
18      FROM STOCKPRICE_ARCHIVE
19  ) s
20  JOIN COMPANY c ON s.company_id = c.company_id;
21  |
22  SELECT * FROM v_company_stock_prices_all
23  WHERE company_id = 1
24  ORDER BY trade_date DESC;
25
```

Script Output ×   ▶ Query Result ×

📌 🖨 📇 📇 SQL | All Rows Fetched: 14 in 0,018 seconds

	COMPANY_ID	COMPANY_NAME	TICKER	TRADE_DATE	OPEN_PRICE	HIGH_PRICE	LOW_PRICE	CLOSE_PRICE	CLOSE_PRICE_PLN	VOLUME
1	1	Amazon.com, Inc.	AMZN	25/04/11	179,93	185,86	178	184,87	780,56	50542500
2	1	Amazon.com, Inc.	AMZN	25/04/10	185,44	186,87	175,85	181,22	765,15	68302000
3	1	Amazon.com, Inc.	AMZN	25/04/10	185,44	186,87	175,85	181,22	765,15	68302000
4	1	Amazon.com, Inc.	AMZN	25/04/09	172,12	192,65	169,93	191,1	806,86	116804300
5	1	Amazon.com, Inc.	AMZN	25/04/09	172,12	192,65	169,93	191,1	806,86	116804300
6	1	Amazon.com, Inc.	AMZN	25/04/08	185,23	185,9	168,57	170,66	720,56	87710400
7	1	Amazon.com, Inc.	AMZN	25/04/08	185,23	185,9	168,57	170,66	720,56	87710400
8	1	Amazon.com, Inc.	AMZN	25/04/07	162	183,41	161,38	175,26	739,98	109327100
9	1	Amazon.com, Inc.	AMZN	25/04/07	162	183,41	161,38	175,26	739,98	109327100
10	1	Amazon.com, Inc.	AMZN	25/04/04	167,15	178,14	166	171	722	122951300

BACKUP

Procedura:

```
create or replace PROCEDURE backup_schema_via_datapump AS
  h1         NUMBER;
  v_jobname  VARCHAR2(50);
  v_dumpfile VARCHAR2(100);
  v_logfile  VARCHAR2(100);
BEGIN
  -- pliki
  v_jobname  := 'SCHEMA_EXPORT_' || TO_CHAR(SYSDATE, 'YYYYMMDD_HH24MISS');
  v_dumpfile := 'backup_' || TO_CHAR(SYSDATE, 'YYYYMMDD_HH24MISS') || '.dmp';
  v_logfile  := 'backup_' || TO_CHAR(SYSDATE, 'YYYYMMDD_HH24MISS') || '.log';

  -- eksport
  h1 := DBMS_DATAPUMP.OPEN(
    operation  => 'EXPORT',
    job_mode   => 'SCHEMA',
    job_name   => v_jobname,
    version    => 'LATEST');

  -- Dump file
  DBMS_DATAPUMP.ADD_FILE(
    handle     => h1,
    filename   => v_dumpfile,
    directory  => 'BACKUP_DIR',
    filetype   => DBMS_DATAPUMP.KU$_FILE_TYPE_DUMP_FILE);

  -- Log file
  DBMS_DATAPUMP.ADD_FILE(
    handle     => h1,
    filename   => v_logfile,
    directory  => 'BACKUP_DIR',
```

```
      filetype   => DBMS_DATAPUMP.KU$_FILE_TYPE_LOG_FILE);

  -- Eksport schematu
  DBMS_DATAPUMP.METADATA_FILTER(
    handle => h1,
    name   => 'SCHEMA_LIST',
    value  => '''KAROL''');

  DBMS_DATAPUMP.START_JOB(h1);
  DBMS_DATAPUMP.DETACH(h1);

  DBMS_OUTPUT.PUT_LINE('Backup zakończony pomyślnie: ' || v_dumpfile);
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Błąd backupu: ' || SQLERRM);
    RAISE;
END;
```

Można odpalić od razu:

```
BEGIN
   backup_schema_via_datapump;
END;
/
```

Lub zaplanować:

```
BEGIN
  DBMS_SCHEDULER.DROP_JOB(
    job_name => 'SCHEMA_BACKUP_JOB',
    force    => TRUE
  );
EXCEPTION
  WHEN OTHERS THEN
    IF SQLCODE != -27475
      RAISE;
    END IF;
END;
/

-- Utworzenie zadania
BEGIN
  DBMS_SCHEDULER.CREATE_JOB(
    job_name        => 'SCHEMA_BACKUP_JOB',
    job_type        => 'PLSQL_BLOCK',
    job_action      => 'BEGIN backup_schema_via_datapump; END;',
    start_date      => SYSTIMESTAMP,
    repeat_interval => 'FREQ=DAILY;BYHOUR=2;BYMINUTE=0;BYSECOND=0',
    enabled         => TRUE,
    comments        => 'Codzienny backup o 02:00'
  );
END;
/

SELECT job_name,
       enabled,
       repeat_interval,
       next_run_date,
       state
  FROM user_scheduler_jobs
 WHERE job_name = 'SCHEMA_BACKUP_JOB';
```

JOB_NAME	ENABLED	REPEAT_INTERVAL	NEXT_RUN_DATE	STATE
1 SCHEMA_BACKUP_JOB	TRUE	FREQ=DAILY;BYHOUR=2;BYMINUTE=0;BYSECOND=0	25/05/10 02:00:00,757012000 +02:00	SCHEDULED

Dodatkowo ustawiono trigger który robi cvs tabeli log, trigger wywołuje procedurę:

```sql
create or replace PROCEDURE export_log_to_csv AS
  v_file  UTL_FILE.FILE_TYPE;
  v_line  VARCHAR2(4000);
  v_message VARCHAR2(4000);
BEGIN
  -- Otwarcie pliku CSV, w BACKUP_DIR
  v_file := UTL_FILE.FOPEN('BACKUP_DIR', 'log_backup.csv', 'W', 4000);

  -- Nagłówek CSV
  UTL_FILE.PUT_LINE(v_file,
'LOG_ID,LOG_TIME,STATUS,OPERATION,USER_NAME,TABLE_NAME,ACTION_DETAIL,MESSAGE');

  -- Iteracja po LOG
  FOR rec IN (
    SELECT LOG_ID,
           LOG_TIME,
           STATUS,
           OPERATION,
           USER_NAME,
           TABLE_NAME,
           ACTION_DETAIL,
           DBMS_LOB.SUBSTR(MESSAGE, 4000, 1) AS MESSAGE
    FROM LOG
    ORDER BY LOG_TIME
  ) LOOP
    -- Zamiana przcinków
    v_line := rec.LOG_ID || ',' ||
              TO_CHAR(rec.LOG_TIME, 'YYYY-MM-DD HH24:MI:SS') || ',' ||
              rec.STATUS || ',' ||
              rec.OPERATION || ',' ||
              rec.USER_NAME || ',' ||
              rec.TABLE_NAME || ',' ||
              rec.ACTION_DETAIL || ',' ||
              rec.MESSAGE;

    UTL_FILE.PUT_LINE(v_file, v_line);
  END LOOP;

  UTL_FILE.FCLOSE(v_file);
  DBMS_OUTPUT.PUT_LINE('Eksport tabeli LOG zakończony powodzeniem.');
EXCEPTION
  WHEN OTHERS THEN
    IF UTL_FILE.IS_OPEN(v_file) THEN
      UTL_FILE.FCLOSE(v_file);
    END IF;
    DBMS_OUTPUT.PUT_LINE('Błąd podczas eksportu: ' || SQLERRM);
    RAISE;
END;
```

Na początek tworzymy folder w serwerze, gdzie będziemy przechowywać pliki kopii:

```
sudo mkdir -p /opt/oracle/backups

sudo chown oracle:oinstall /opt/oracle/backups

sudo chmod 775 /opt/oracle/backups
```

Jako SYS:

```
CREATE OR REPLACE DIRECTORY BACKUP_DIR AS '/opt/oracle/backups';

GRANT READ, WRITE ON DIRECTORY BACKUP_DIR TO KAROL;

GRANT READ, WRITE ON DIRECTORY BACKUP_DIR TO SYSTEM;

GRANT READ, WRITE ON DIRECTORY BACKUP_DIR TO SYSTEM;

GRANT READ, WRITE ON DIRECTORY BACKUP_DIR TO TEST;
```

Po utworzeniu pliku dmp można zimportować plik bazy.

VIEW->DBA ->NEW Connection ->jako SYSTEM->Data Pump->Data Pump Import Wizard

Remapping

- Type
- Filter
- **Remapping**
- Options
- Schedule
- Summary

Re-Map Schemas

Source	Destination
KAROL	TEST4

Add Row   Remove Row

Re-Map Tablespaces

Source	Destination
KAROL_TS	TEST4_TBS

Add Row   Remove Row

Re-Map Datafiles

Source	Destination

Add Row   Remove Row

Pomoc   < Wstecz   Dalej >   Zakończ   Anuluj

Summary

- Type
- Filter
- Remapping
- Options
- Schedule
- **Summary**

| Summary | PL/SQL |

- Data Pump Summary
  - Connections
  - Data
  - Options

Setting up Data Pump job.

Setting up Data Pump job.

Processing...

Pomoc   < Wstecz   Dalej >   Zakończ   Anuluj

≡ STOCK APP V2

STOCK APP V2

Firmy
- 🏛 Firmy (Tabela)
- ➕ Dodaj Firmę

Inwestorzy
- 👫 Inwestorzy (Tabela)
- ➕ Dodaj Inwestora

Transakcje
- 🎰 Pokaż transakcje
- 💰 Wykonaj transakcję

LOG
- 📋 Wyświetl LOG

Raporty
- 📑 Wyświetl raporty
- ➕ Utwórz raport

Widoki
- 👁 Firmy (Pochodzenie)
- 👁 Firmy (Liczba akcji)

Home \

Investor_Page

		Client Code	Name	Email	Phone	National Id	Created At
🗑	✏	AA113	Robert Kubica	robert@f1.com	+4455533222	15345678925	4/8/2025
🗑	✏	CL9999	Jan Kowalski	jan.kowalski@sql.com	+48123456789	85010112345	4/12/2025
🗑	✏	CL111	Jan Kowalski	jan.kowalski@f1.com	+48123456789	12345678901	4/3/2025
🗑	✏	CL222	Anna Nowak	anna.nowak@sql.com	+48123456780	23456789012	4/3/2025
🗑	✏	CL003	Piotr Wiśniewski	piotr.wisniewski@sql.com	+48123456781	34567890123	4/3/2025
🗑	✏	CL004	Katarzyna Zielińska	katarzyna.zielinska@sql.com	+48123456782	45678901234	4/3/2025
🗑	✏	CL005	Tomasz Wójcik	tomasz.wojcik@sql.com	+48123456783	56789012345	4/3/2025

1 - 7

◀◀ Cofnij

InvestorDelete_Page

Usuwanie Inwestora

Client Code
AA113

Name
Robert Kubica

Email
robert@f1.com

Phone
+4455533222

National Id
15345678925

◀◀ Anuluj   🗑 Usuń Inwestora

Company_Page

		Name	Ticker	Sector	Country	Website
🗑	✎	Adax	ADX	Banking	Poland	www.adax.uk
🗑	✎	Amazon.com, Inc.	AMZN	Consumer Services	USA	http://amazon.com
🗑	✎	Alphabet Inc.	GOOGL	Technology	USA	http://abc.xyz
🗑	✎	Meta Platforms, Inc.	META	Technology	USA	http://facebook.com
🗑	✎	Alibaba Group Holding Ltd.	BABA	Consumer Services	China	http://alibaba.com
🗑	✎	Johnson & Johnson	JNJ	Health Care	USA	http://jnj.com
🗑	✎	JPMorgan Chase & Co.	JPM	Financials	USA	http://jpmorganchase.com
🗑	✎	Visa Inc.	V	Financials	USA	http://visa.com
🗑	✎	Procter & Gamble Co.	PG	Consumer Goods	USA	http://pg.com
🗑	✎	NVIDIA Corporation	NVDA	Technology	USA	http://nvidia.com
🗑	✎	Toyota Motor Corporation	TM	Consumer Goods	Japan	http://toyota-global.com
🗑	✎	NestlÃ© S.A.	NESN.SW	Consumer Goods	Switzerland	http://nestle.com
🗑	✎	Roche Holding AG	ROG.SW	Health Care	Switzerland	http://roche.com
🗑	✎	HSBC Holdings plc	HSBC	Financials	UK	http://hsbc.com
🗑	✎	TotalEnergies SE	TTE	Energy	France	http://totalenergies.com

1 - 15   Next ▶

◀◀ Cofnij

CompanyEdit

Edytowanie Firmy

Name
Amazon.com, Inc.

New
AMZN

New
Consumer Services

New
USA

New
http://amazon.com

◁◁ Anuluj   ✐ Edytuj Firmę

PickPeriod_Page

Tabela transakcji

Investor Id ↑≡	Transaction Id	Company Id	Operation	Shares	Price Per Share	Commission Rate	Total Value	Transaction Date
1	2	5	BUY	100	150.25	0	15100.13	4/12/2025
1	1	15	BUY	5	115.38	7	617.28	4/1/2025
1	105	13	BUY	66	415.45	5	28790.69	2/11/2025
1	106	11	BUY	95	271.57	3	26573.12	2/12/2025
1	108	4	BUY	45	507.94	3	23543.02	2/13/2025
1	117	9	SELL	95	290.31	3	28406.83	1/31/2025
1	119	15	SELL	42	180.57	0	7583.94	2/8/2025
1	123	12	SELL	71	580.7	0	41229.7	4/27/2025
1	127	13	SELL	93	52.38	7	5212.33	4/20/2025
1	171	3	BUY	37	504.11	7	19957.71	4/19/2025
1	181	4	SELL	46	389.18	5	18797.39	4/7/2025
1	183	1	SELL	87	174.66	3	15651.28	2/4/2025
1	185	4	BUY	1	130.07	3	133.97	1/27/2025
1	193	4	SELL	75	273.99	0	20549.25	4/24/2025
1	211	15	BUY	63	528.55	3	34297.61	4/24/2025

1 - 15   Next ▶

◁◁ Cofnij

Formularz transakcji

Numer Inwestora
AA113 - Robert Kubica ⌄

Firma
ADX - Adax ⌄

Cena Akcji
18.35

Liczba akcji
4

Typ operacji
KUP ⌄

Prowizja (%)
4

Wartość w $
0

Wartość w zł
0

🜂 **Wykonaj transakcję**

◁◁ Cofnij

Home \

LOG_Page

LOG

Log Id ↑≡	Log Time	Status	Operation	User Name	Table Name	Action Detail	Message
669	4/12/2025	ERROR	INSERT	KAROL	INVESTOR	add_investor	ORA-20002: Niepoprawny numer telefonu ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 25 ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
670	4/12/2025	ERROR	INSERT	KAROL	INVESTOR	add_investor	ORA-20003: Niepoprawny numer identyfikacyjny (PESEL) ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 27 ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
671	4/12/2025	ERROR	INSERT	KAROL	INVESTOR	add_investor	ORA-20001: Niepoprawny adres e-mail ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 23 ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
672	4/12/2025	OK	INSERT	KAROL	INVESTOR	add_investor	Dodano inwestora: Jan Kowalski
691	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-20006: Nieznana operacja. Dozwolone: BUY lub SELL.
692	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-01400: cannot insert NULL into ("KAROL"."PORTFOLIO"."INVESTOR_ID") ORA-01403: no data found
693	4/12/2025	OK	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	Transakcja: BUY 10 akcji
711	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-20006: Nieznana operacja. Dozwolone: BUY lub SELL.
712	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-01400: cannot insert NULL into ("KAROL"."PORTFOLIO"."INVESTOR_ID") ORA-01403: no data found
713	4/12/2025	OK	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	Transakcja: BUY 50 akcji
714	4/12/2025	OK	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	Transakcja: BUY 50 akcji
715	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-20006: Nieznana operacja. Dozwolone: BUY lub SELL.
716	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-01400: cannot insert NULL into ("KAROL"."PORTFOLIO"."INVESTOR_ID") ORA-01403: no data found
717	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-01400: cannot insert NULL into ("KAROL"."PORTFOLIO"."INVESTOR_ID") ORA-01403: no data found

Home \

RaportForm_Page

Generowanie raportu

| Inwestor | ⌄ | | Zakres
Ostatni rok | ⌄ |

SQL

```
BEGIN
   generate_activity_summary(
      p_investor_id => :P14_NAME,
      p_period_type => :P14_PERIOD
   );
END;
```

◄◄ Cofnij   🗒 Wygeneruj raport

Home \ RaportForm_Page \

Raport_Page

Summary Id	Investor ID	Period Type	Total Value Pln	Generated At ↑≡	Period From	Period To	Number Of Transactions
2	1	MONTH	2,606.3	4/12/2025	3/13/2025	4/12/2025	1
3	2	MONTH	120,129.0	4/12/2025	3/13/2025	4/12/2025	5
4	2	QUARTER	120,129.0	4/12/2025	1/12/2025	4/12/2025	5
5	4	MONTH	8,481.6	4/12/2025	3/13/2025	4/12/2025	2
21	2	QUARTER	1,321,457.2	4/12/2025	1/12/2025	4/12/2025	20
22	8	QUARTER	910,696.4	4/12/2025	1/12/2025	4/12/2025	20
41	4	YEAR	1,486,939.4	4/12/2025	4/12/2024	4/12/2025	27
42	8	YEAR	1,089,016.3	4/12/2025	4/12/2024	4/12/2025	24

◄◄ Cofnij

Home \ RaportForm_Page \ Raport_Page \

RaportOverview

Nr raportu
21

Kod klienta
CL222

Inwestor
Anna Nowak

Od
1/12/2025

Do
4/12/2025

Liczba wykonanych transakcji
20

Wartość
1321457.15

◁◁ Cofnij

Home \

LOG_Page

LOG

Log Id ↑≡	Log Time	Status	Operation	User Name	Table Name	Action Detail	Message
669	4/12/2025	ERROR	INSERT	KAROL	INVESTOR	add_investor	ORA-20002: Niepoprawny numer telefonu ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 25 ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
670	4/12/2025	ERROR	INSERT	KAROL	INVESTOR	add_investor	ORA-20003: Niepoprawny numer identyfikacyjny (PESEL) ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 27 ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
671	4/12/2025	ERROR	INSERT	KAROL	INVESTOR	add_investor	ORA-20001: Niepoprawny adres e-mail ORA-06512: przy "KAROL.TRG_VALIDATE_INVESTOR", linia 23 ORA-04088: błąd w trakcie wykonywania wyzwalacza 'KAROL.TRG_VALIDATE_INVESTOR'
672	4/12/2025	OK	INSERT	KAROL	INVESTOR	add_investor	Dodano inwestora: Jan Kowalski
691	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-20006: Nieznana operacja. Dozwolone: BUY lub SELL.
692	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-01400: cannot insert NULL into ("KAROL"."PORTFOLIO"."INVESTOR_ID") ORA-01403: no data found
693	4/12/2025	OK	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	Transakcja: BUY 10 akcji
711	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-20006: Nieznana operacja. Dozwolone: BUY lub SELL.
712	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-01400: cannot insert NULL into ("KAROL"."PORTFOLIO"."INVESTOR_ID") ORA-01403: no data found
713	4/12/2025	OK	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	Transakcja: BUY 50 akcji
714	4/12/2025	OK	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	Transakcja: BUY 50 akcji
715	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-20006: Nieznana operacja. Dozwolone: BUY lub SELL.
716	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-01400: cannot insert NULL into ("KAROL"."PORTFOLIO"."INVESTOR_ID") ORA-01403: no data found
717	4/12/2025	ERROR	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	ORA-01400: cannot insert NULL into ("KAROL"."PORTFOLIO"."INVESTOR_ID") ORA-01403: no data found
718	4/12/2025	OK	INSERT	APEX_PUBLIC_USER	TRANSACTION	make_transaction	Transakcja: BUY 10 akcji

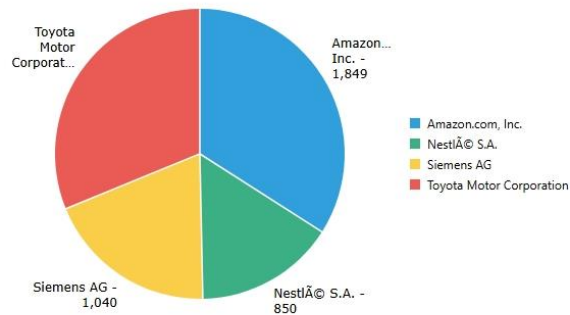# InvestorPorfolio_Page

Portfolio inwestora:

Client Code	Name
CL005	Tomasz Wójcik

Posiadane akcje

| Firma ↑≞	Liczba Akcji	Wartość [$]
Amazon.com, Inc.	10	1848.7$
NestlÃ© S.A.	10	849.8$
Siemens AG	10	1040$
Toyota Motor Corporation	10	1694.2$

1 - 4

Portfel

SQL

SELECT COMPANY.NAME AS Firma, PORTFOLIO.SHARES AS "Liczba akcji", CONCAT(STOCKPRICE.CLOSE_PRICE * PORTFOLIO.SHARES, '$') AS "Wartość [$]" FROM PORTFOLIO INNER JOIN COMPANY ON COMPANY.COMPANY_ID=PORTFOLIO.COMPANY_ID INNER JOIN STOCKPRICE ON STOCKPRICE.COMPANY_ID = PORTFOLIO.COMPANY_ID WHERE PORTFOLIO.INVESTOR_ID = :P12_INVESTOR_ID;

◁◁ Cofnij

☰ STOCK APP V2

CompanyOrigin_view

Widok - Firmy (Pochodzenie)
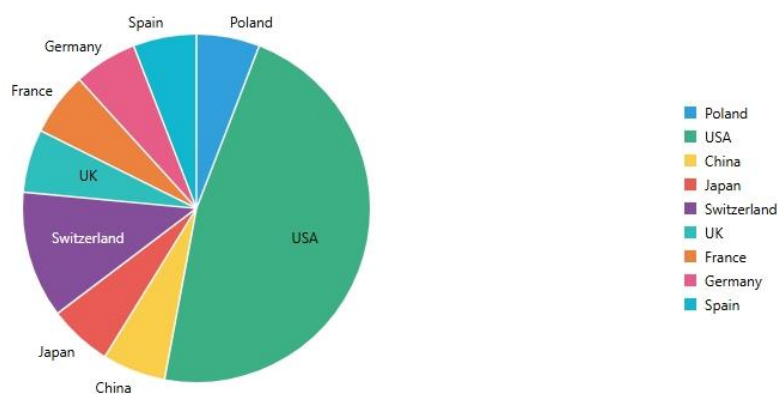
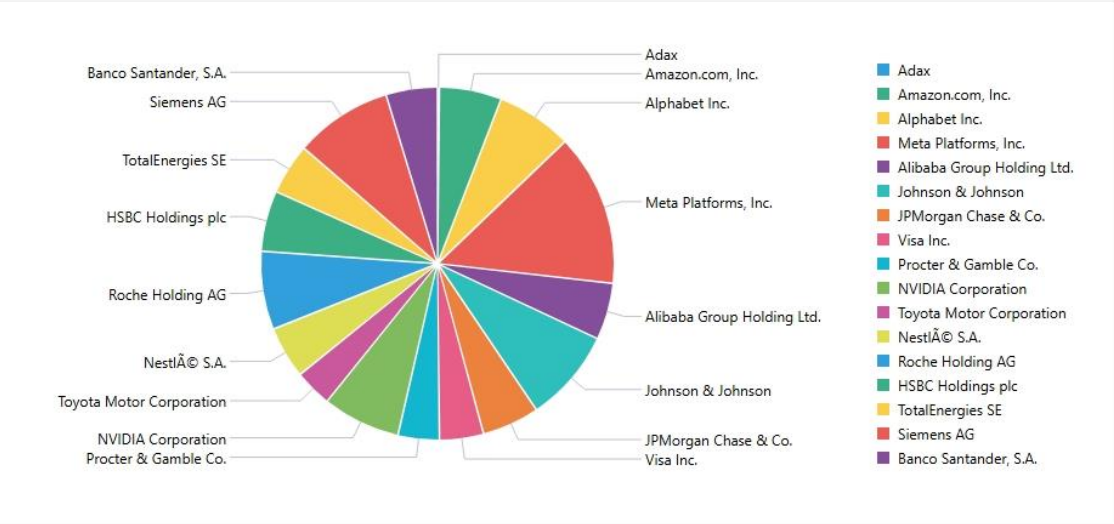| Country ↑≞	Suma
China	1
France	1
Germany	1
Japan	1
Poland	1
Spain	1
Switzerland	2
UK	1
USA	8

1 - 9

SQL

CREATE VIEW v_CompanyOrigin AS SELECT COUNTRY, COUNT(*) as SUMA FROM COMPANY GROUP BY COUNTRY;

◁◁ Cofnij

Home \

CompanyShares_view

Widok - Firmy (Pochodzenie)

Name ↑≡	Akcje
Adax	14
Alibaba Group Holding Ltd.	428
Alphabet Inc.	572
Amazon.com, Inc.	468
Banco Santander, S.A.	387
HSBC Holdings plc	456
JPMorgan Chase & Co.	432
Johnson & Johnson	702
Meta Platforms, Inc.	1147
NVIDIA Corporation	587
Nestlé S.A.	392
Procter & Gamble Co.	311
Roche Holding AG	590
Siemens AG	734
TotalEnergies SE	384

1 - 15   Next ▸

SQL

CREATE VIEW v_CompanyShares **AS SELECT** c.Name, **SUM(**t.Shares**) as** AKCJE **FROM** TRANSACTION t **JOIN** COMPANY c **ON** t.Company_ID = c.Company_ID **GROUP BY** c.NAME;