

PortFolioIQ: Financial Portfolio Management System

Milestone 2

Of

Data Warehousing and Integration

BY

GROUP NUMBER 14:

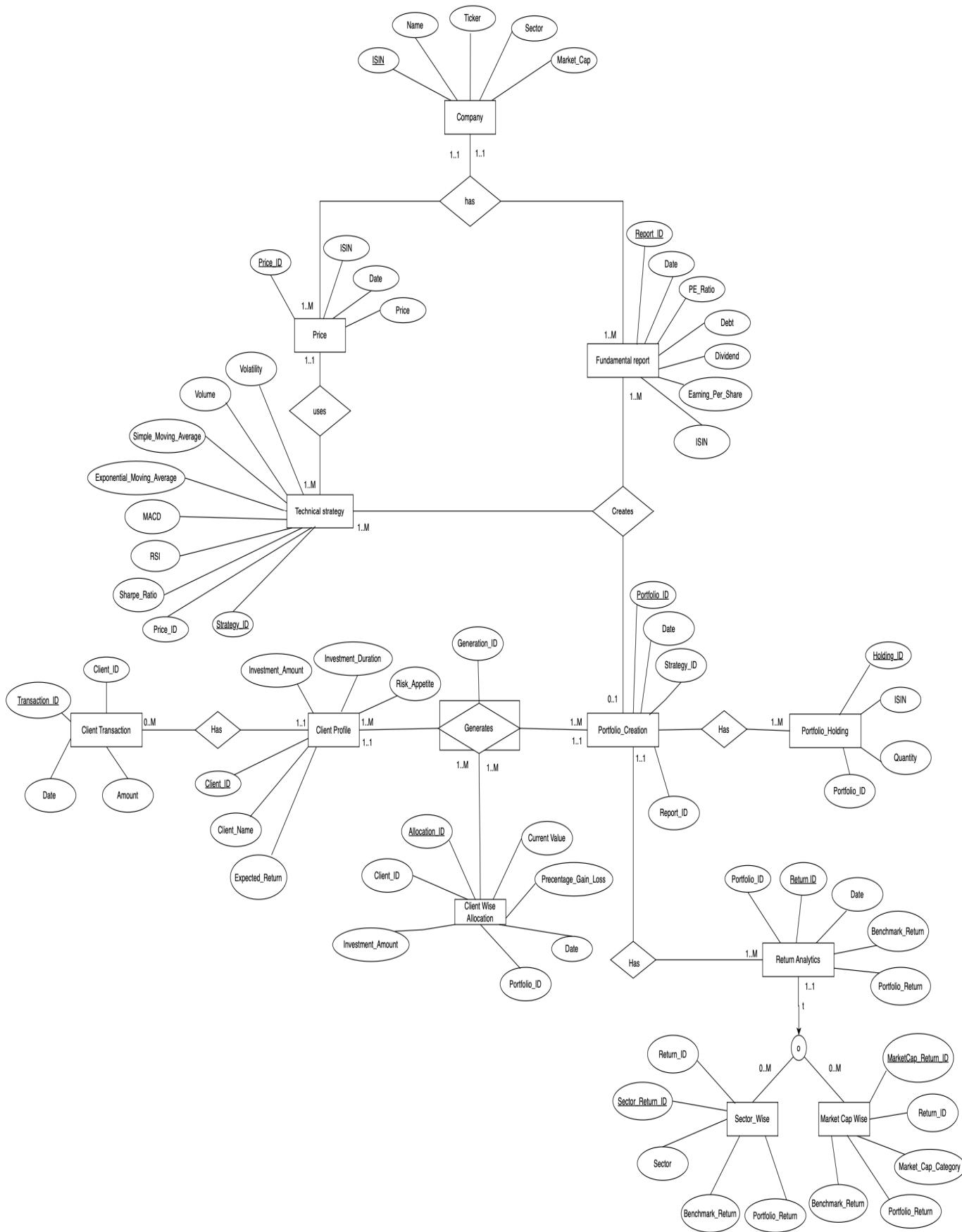
ARCHIT SINGH (002813253)

SANCIA SEROPHENE SALDANHA (002851577)

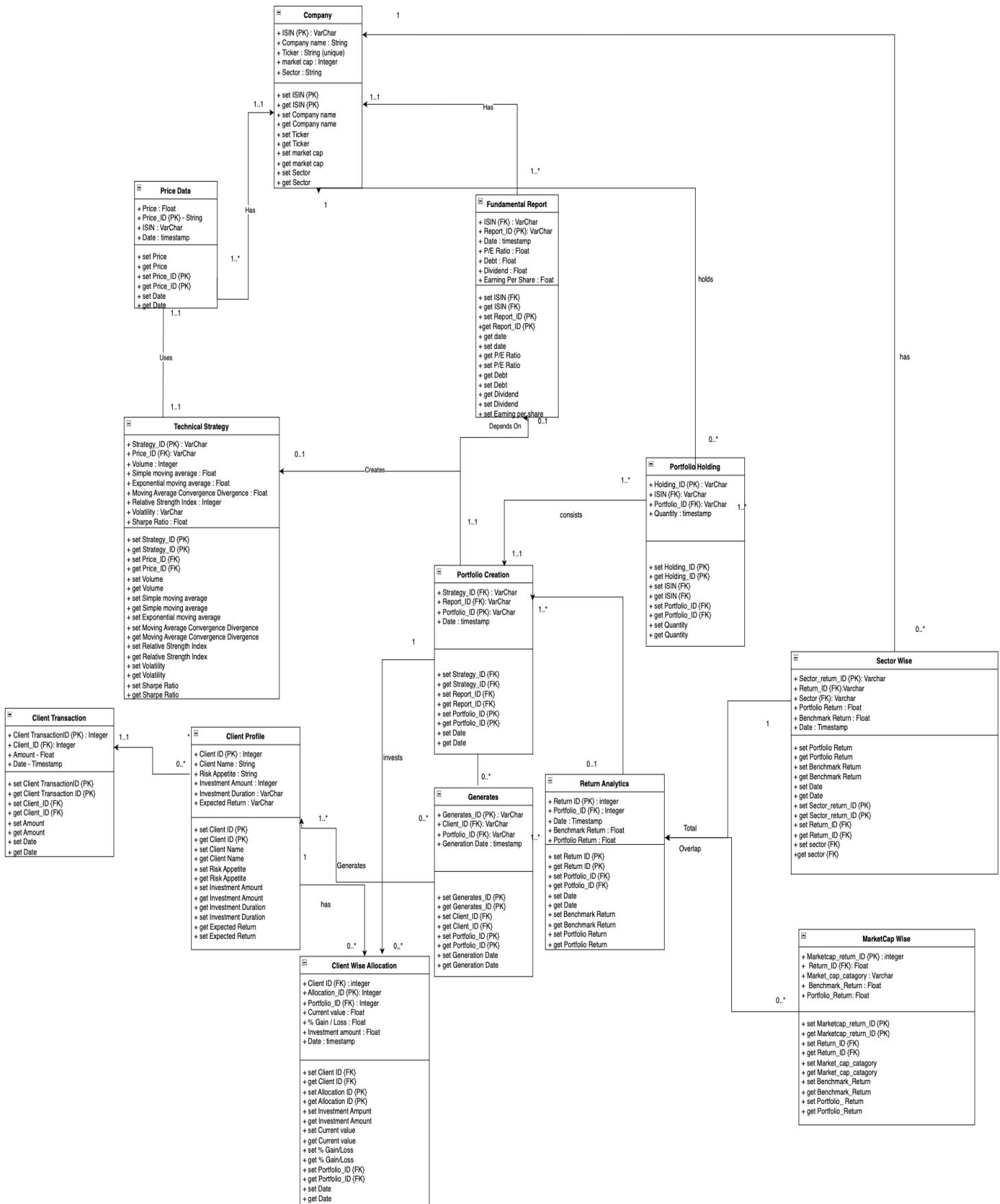


**DEPARTMENT OF COLLEGE OF ENGINEERING
NORTHEASTERN UNIVERSITY
BOSTON, MASSACHUSETTS – 022115**

A. Entity-Relationship Diagram:



B. Relational Model:



C. NORMALIZED RELATIONAL MODEL:

1. Company (ISIN, Ticker, Name, Sector, Market_Cap)

Primary Key:

ISIN (NOT NULL)

Attributes:

Ticker (NOT NULL)

Name

Sector

Market_Cap

2. Price (Price_ID, ISIN, Date, Price)

Primary Key:

Price_ID (NOT NULL)

Foreign Key:

ISIN references Company(ISIN) (NOT NULL)

Attributes:

Date (NOT NULL)

Price (NOT NULL)

3. Fundamental_Report (Report_ID, ISIN, Date, PE_Ratio, Debt, Dividend, Earnings_Per_Share)

Primary Key:

Report_ID (NOT NULL)

Foreign Key:

ISIN references Company(ISIN) (NOT NULL)

Attributes:

Date (NOT NULL)

PE_Ratio

Debt

Dividend

Earnings_Per_Share

4. Technical_Strategy (Strategy_ID, Price_ID, Volatility, Volume, Simple_Moving_Average, Exponential_Moving_Average, MACD, RSI, Sharpe_Ratio)

Primary Key:

Strategy_ID (NOT NULL)

Foreign Key:

Price_ID references Price(Price_ID) (NOT NULL)

Attributes:

Volatility

Volume

Simple_Moving_Average

Exponential_Moving_Average

MACD

RSI

Sharpe_Ratio

5. Portfolio_Creation (Portfolio_ID, Strategy_ID, Report_ID, Date)

Primary Key:

Portfolio_ID (NOT NULL)

Foreign Keys:

Strategy_ID references Technical_Strategy(Strategy_ID) (NULLABLE)

Report_ID references Fundamental_Report(Report_ID) (NULLABLE)

Attributes:

Date (NOT NULL)

6. Portfolio_Holding (Holding_ID, Portfolio_ID, ISIN, Quantity)

Primary Key:

Holding_ID (NOT NULL)

Foreign Keys:

Portfolio_ID references Portfolio_Creation(Portfolio_ID) (NOT NULL)

ISIN references Company(ISIN) (NOT NULL)

Attributes:

Quantity (NOT NULL)

7. Client_Profile (Client_ID, Client_Name, Expected_Return, Investment_Amount, Investment_Duration, Risk_Appetite)

Primary Key:

Client_ID (NOT NULL)

Attributes:

Client_Name

Expected_Return

Investment_Amount

Investment_Duration

Risk_Appetite

8. Generates (Generation_ID, Client_ID, Portfolio_ID, Generation_Date)

Primary Key:

Generation_ID (NOT NULL)

Foreign Keys:

Client_ID references Client_Profile(Client_ID) (NOT NULL)

Portfolio_ID references Portfolio_Creation(Portfolio_ID) (NOT NULL)

Attributes:

Generation_Date (NOT NULL)

9. Client_Transaction (Transaction_ID, Client_ID, Date, Amount)

Primary Key:

Transaction_ID (NOT NULL)

Foreign Key:

Client_ID references Client_Profile(Client_ID) (NOT NULL)

Attributes:

Date (NOT NULL)

Amount (NOT NULL)

- 10. Client_Wise_Allocation (Allocation_ID, Client_ID, Portfolio_ID, Date, Investment_Amount, Current_Value, Percentage_Gain_Loss)**

Primary Key:

Allocation_ID (NOT NULL)

Foreign Keys:

Client_ID references Client_Profile(Client_ID) (NOT NULL)

Portfolio_ID references Portfolio_Creation(Portfolio_ID) (NOT NULL)

Attributes:

Date (NOT NULL)

Investment_Amount

Current_Value

Percentage_Gain_Loss

- 11. Return_Analytics (Return_ID, Portfolio_ID, Date, Benchmark_Return, Portfolio_Return)**

Primary Key:

Return_ID (NOT NULL)

Foreign Key:

Portfolio_ID references Portfolio_Creation(Portfolio_ID) (NOT NULL)

Attributes:

Date (NOT NULL)

Benchmark_Return

Portfolio_Return

- 12. Sector_Wise (Sector_Return_ID, Return_ID, Sector, Benchmark_Return, Portfolio_Return)**

Primary Key:

Sector_Return_ID (NOT NULL)

Foreign Keys:

Return_ID references Return_Analytics(Return_ID) (NOT NULL)

Sector references Company(Sector) (NOT NULL)

Attributes:

Benchmark_Return

Portfolio_Return

- 13. Market_Cap_Wise (MarketCap_Return_ID, Return_ID, Market_Cap_Category, Benchmark_Return, Portfolio_Return)**

Primary Key:

MarketCap_Return_ID (NOT NULL)

Foreign Key:

Return_ID references Return_Analytics(Return_ID) (NOT NULL)

Attributes:

Market_Cap_Category (NOT NULL)

Benchmark_Return

Portfolio_Return

D. POTENTIAL DIMENSIONS, HIERARCHIES AND MEASURES:

Dimensions:

- **Clients:** The individuals who invest in portfolios, including their names, expected returns, investment amounts, durations, risk appetites, and unique IDs.
- **Portfolios:** Collections of investments created using specific strategies and reports, detailing when they were created, which strategies and reports were used, and unique portfolio IDs.
- **Companies:** The businesses available for investment, with information like their names, ticker symbols, sectors, market capitalizations, and unique identifiers (ISINs).
- **Holdings:** The specific investments within each portfolio, including which companies are held, the quantities owned, and unique holding IDs.
- **Transactions:** Records of financial activities made by clients, such as deposits and withdrawals, including dates, amounts, and unique transaction IDs.
- **Strategies and Reports:** The methods and analyses used to create portfolios, including technical strategies (like volatility and moving averages), fundamental reports (like P/E ratios and earnings per share), and their unique IDs.
- **Sectors:** Categories grouping companies based on the industries they operate in, such as Technology, Healthcare, or Finance.
- **Market Cap Categories:** Classifications of companies based on their market capitalization, like Mega Cap, Large Cap, Mid Cap, and Small Cap.
- **Time:** Information about when events occur, including dates, months, quarters, and years, allowing for time-based analysis.

Hierarchies:

- **Client Hierarchy:** Imagine a tree starting with a client at the root, branching out to the portfolios they've generated, then to the holdings within those portfolios, and finally to the companies they've invested in, including the sectors and market cap categories of those companies.
- **Portfolio Hierarchy:** Picture another tree beginning with a portfolio, branching into its holdings, then to the companies included, and further into their sectors and market cap categories.
- **Company Hierarchy:** Envision a structure starting with a company, then branching into its sector and market cap category, helping to categorize investments.
- **Time Hierarchy:** Think of a timeline that starts with the year, then narrows down to quarters, months, and days, helping to analyze trends over time.

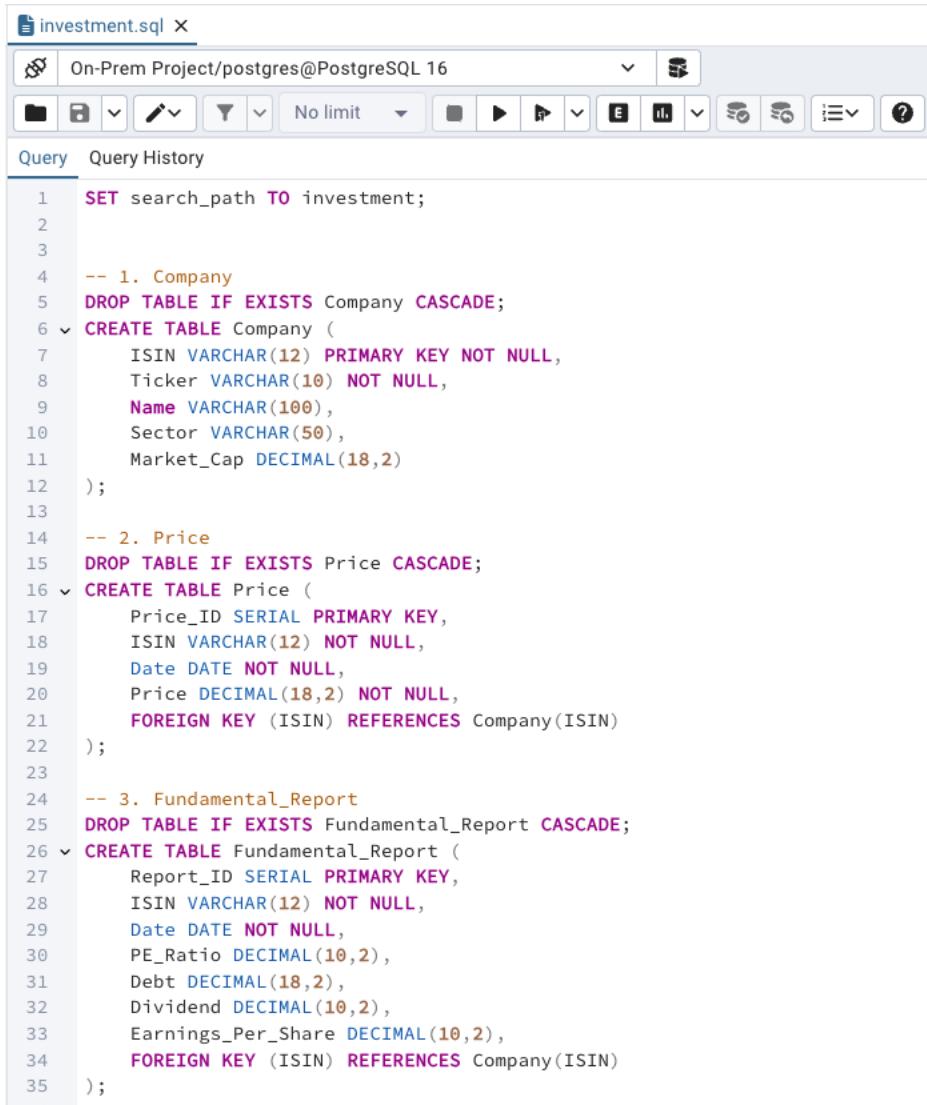
Measures:

- **Investments:** This includes the total amount of money clients have invested, the current value of their investments, and the percentage gain or loss they've experienced.

- **Returns:** The performance metrics of portfolios, such as how much return a portfolio has generated compared to benchmarks, and returns analyzed by sectors or market cap categories.
- **Transactions:** We can count and sum the transactions clients make to understand their activity levels, including how much they're depositing or withdrawing over time.
- **Technical Indicators:** Measures like volatility, volume, moving averages, MACD, RSI, and Sharpe Ratio that help assess the performance and risk of investments.
- **Fundamental Metrics:** Financial indicators like P/E ratios, debt levels, dividends, and earnings per share that provide insights into a company's financial health.
- **Client Activity:** The number of portfolios each client has, their investment durations, and their risk appetites can be measured to understand client behaviors.
- **Market Performance:** Tracking stock prices and price changes over time to gauge market trends.

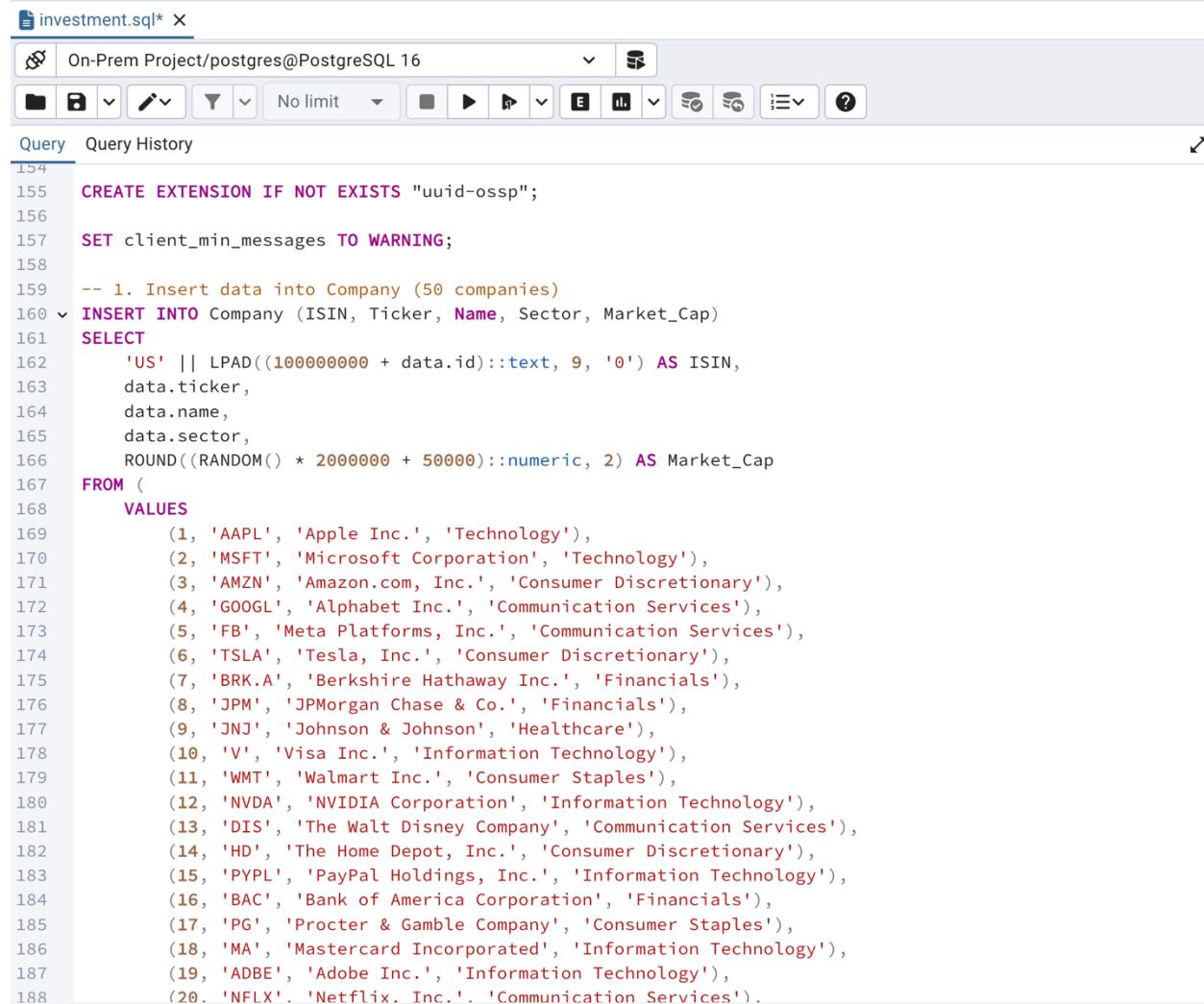
E. IMPLEMENTATION OF RELATIONAL MODEL IN POSTGRES:

Initially, we created a table schema in PostgreSQL:



```
investment.sql X
On-Prem Project/postgres@PostgreSQL 16
Query History
1 SET search_path TO investment;
2
3
4 -- 1. Company
5 DROP TABLE IF EXISTS Company CASCADE;
6 CREATE TABLE Company (
7     ISIN VARCHAR(12) PRIMARY KEY NOT NULL,
8     Ticker VARCHAR(10) NOT NULL,
9     Name VARCHAR(100),
10    Sector VARCHAR(50),
11    Market_Cap DECIMAL(18,2)
12 );
13
14 -- 2. Price
15 DROP TABLE IF EXISTS Price CASCADE;
16 CREATE TABLE Price (
17     Price_ID SERIAL PRIMARY KEY,
18     ISIN VARCHAR(12) NOT NULL,
19     Date DATE NOT NULL,
20     Price DECIMAL(18,2) NOT NULL,
21     FOREIGN KEY (ISIN) REFERENCES Company(ISIN)
22 );
23
24 -- 3. Fundamental_Report
25 DROP TABLE IF EXISTS Fundamental_Report CASCADE;
26 CREATE TABLE Fundamental_Report (
27     Report_ID SERIAL PRIMARY KEY,
28     ISIN VARCHAR(12) NOT NULL,
29     Date DATE NOT NULL,
30     PE_Ratio DECIMAL(10,2),
31     Debt DECIMAL(18,2),
32     Dividend DECIMAL(10,2),
33     Earnings_Per_Share DECIMAL(10,2),
34     FOREIGN KEY (ISIN) REFERENCES Company(ISIN)
35 );
```

We then make use of the extension “uuid-ossp” to generate the data for all the tables:



```
investment.sql* ×
On-Prem Project/postgres@PostgreSQL 16
No limit ▾ E ⌂ ⌂ ⓘ ? ↗

Query History
154
155 CREATE EXTENSION IF NOT EXISTS "uuid-ossp";
156
157 SET client_min_messages TO WARNING;
158
159 -- 1. Insert data into Company (50 companies)
160 ✓ INSERT INTO Company (ISIN, Ticker, Name, Sector, Market_Cap)
161 SELECT
162     'US' || LPAD((100000000 + data.id)::text, 9, '0') AS ISIN,
163     data.ticker,
164     data.name,
165     data.sector,
166     ROUND((RANDOM() * 2000000 + 50000)::numeric, 2) AS Market_Cap
167 FROM (
168     VALUES
169         (1, 'AAPL', 'Apple Inc.', 'Technology'),
170         (2, 'MSFT', 'Microsoft Corporation', 'Technology'),
171         (3, 'AMZN', 'Amazon.com, Inc.', 'Consumer Discretionary'),
172         (4, 'GOOGL', 'Alphabet Inc.', 'Communication Services'),
173         (5, 'FB', 'Meta Platforms, Inc.', 'Communication Services'),
174         (6, 'TSLA', 'Tesla, Inc.', 'Consumer Discretionary'),
175         (7, 'BRK.A', 'Berkshire Hathaway Inc.', 'Financials'),
176         (8, 'JPM', 'JPMorgan Chase & Co.', 'Financials'),
177         (9, 'JNJ', 'Johnson & Johnson', 'Healthcare'),
178         (10, 'V', 'Visa Inc.', 'Information Technology'),
179         (11, 'WMT', 'Walmart Inc.', 'Consumer Staples'),
180         (12, 'NVDA', 'NVIDIA Corporation', 'Information Technology'),
181         (13, 'DIS', 'The Walt Disney Company', 'Communication Services'),
182         (14, 'HD', 'The Home Depot, Inc.', 'Consumer Discretionary'),
183         (15, 'PYPL', 'PayPal Holdings, Inc.', 'Information Technology'),
184         (16, 'BAC', 'Bank of America Corporation', 'Financials'),
185         (17, 'PG', 'Procter & Gamble Company', 'Consumer Staples'),
186         (18, 'MA', 'Mastercard Incorporated', 'Information Technology'),
187         (19, 'ADBE', 'Adobe Inc.', 'Information Technology'),
188         (20, 'NFLX', 'Netflix. Inc.', 'Communication Services').
```

As displayed the data has been successfully generated for each of the tables:

1. Company:

The screenshot shows the pgAdmin interface with the Object Explorer on the left and a query editor on the right. The query editor contains the SQL code for creating the Company table and executing a select statement to view its data.

```

SET search_path TO investment;
-- 1. Company
DROP TABLE IF EXISTS Company CASCADE;
CREATE TABLE Company (
    ISIN VARCHAR(12) PRIMARY KEY NOT NULL,
    Ticker VARCHAR(10) NOT NULL,
    Name VARCHAR(100),
    Sector VARCHAR(50),
    Market_Cap DECIMAL(18,2)
);
Select * From Company;

```

The Data Output tab shows the results of the query, displaying 60 rows of company data. The columns are: isin [PK] character varying (12), ticker character varying (10), name character varying (100), sector character varying (50), and market_cap numeric (18,2). The data includes various well-known companies like Intel, Comcast, Verizon, AT&T, Cisco, Pepsico, The Coca-Cola Company, Oracle, Abbott Laboratories, Salesforce, Nike, Merck, Costco, Danaher, Wells Fargo, McDonald's, Medtronic, Accenture, and Texas Instruments.

isIn	Ticker	Name	Sector	Market_Cap
US100000021	INTC	Intel Corporation	Information Technology	1919353.85
US100000022	CMCSA	Comcast Corporation	Communication Services	105776.31
US100000023	VZ	Verizon Communications Inc.	Communication Services	77958.64
US100000024	T	AT&T Inc.	Communication Services	1649317.40
US100000025	CSCO	Cisco Systems, Inc.	Information Technology	592591.26
US100000026	PEP	PepsiCo, Inc.	Consumer Staples	1745460.95
US100000027	KO	The Coca-Cola Company	Consumer Staples	1675606.05
US100000028	ORCL	Oracle Corporation	Information Technology	1902758.08
US100000029	ABT	Abbott Laboratories	Healthcare	1431909.57
US100000030	CRM	Salesforce.com, Inc.	Information Technology	1555807.69
US100000031	NKE	Nike, Inc.	Consumer Discretionary	1111449.55
US100000032	MRK	Merck & Co., Inc.	Healthcare	260766.90
US100000033	COST	Costco Wholesale Corporation	Consumer Staples	433721.89
US100000034	DHR	Danaher Corporation	Healthcare	1555613.22
US100000035	WFC	Wells Fargo & Company	Financials	603815.73
US100000036	MCD	McDonald's Corporation	Consumer Discretionary	759304.07
US100000037	MDT	Medtronic plc	Healthcare	1973273.20
US100000038	ACN	Accenture plc	Information Technology	1249173.76
US100000039	TXN	Texas Instruments Incorporated	Information Technology	1525489.60

2. Price:

The screenshot shows the pgAdmin interface with the Object Explorer on the left and a query editor on the right. The query editor contains the SQL code for creating the Price table and executing a select statement to view its data.

```

-- 2. Price
DROP TABLE IF EXISTS Price CASCADE;
CREATE TABLE Price (
    price_id SERIAL PRIMARY KEY,
    ISIN VARCHAR(12) NOT NULL,
    Date DATE NOT NULL,
    Price DECIMAL(18,2) NOT NULL,
    FOREIGN KEY (ISIN) REFERENCES Company(ISIN)
);
Select * From Price;

```

The Data Output tab shows the results of the query, displaying 100 rows of price data. The columns are: price_id [PK] integer, isin character varying (12), date date, and price numeric (18,2). The data shows price points for various companies on specific dates, such as Intel at \$1773.62 on 2024-10-03, Comcast at \$1711.66 on the same date, and so on for other companies like Verizon, AT&T, Cisco, etc.

price_id	isin	date	price
1	US0100000001	2024-10-03	1773.62
2	US0100000002	2024-10-03	1711.66
3	US0100000003	2024-10-03	1921.59
4	US0100000004	2024-10-03	228.98
5	US0100000005	2024-10-03	1357.02
6	US0100000006	2024-10-03	391.33
7	US0100000007	2024-10-03	649.60
8	US0100000008	2024-10-03	2019.66
9	US0100000009	2024-10-03	964.41
10	US0100000010	2024-10-03	61.25
11	US0100000001	2024-10-02	1467.36
12	US0100000002	2024-10-02	673.21
13	US0100000003	2024-10-02	799.92
14	US0100000004	2024-10-02	1692.08
15	US0100000005	2024-10-02	243.83
16	US0100000006	2024-10-02	518.60
17	US0100000007	2024-10-02	1903.69
18	US0100000008	2024-10-02	1438.04
19	US0100000009	2024-10-02	1171.09

3. Fundamental_Report:

Object Explorer

```

    \v Servers (1)
      \v PostgreSQL 16
        \v Databases (3)
          > Northwind
          > On-Prem Project
            > Casts
            > Catalogs
            > Event Triggers
            > Extensions
            > Foreign Data Wrappers
            > Languages
            > Publications
          \v Schemas (2)
            \diamond Investment
              > Aggregates
              > Collations
              > Domains
              > FTS Configurations
              > FTS Dictionaries
              > FTS Parsers
              > FTS Templates
              > Foreign Tables
              > Functions
              > Materialized Views
              > Operators
              > Procedures
              > Sequences
            \v Tables (13)
              > client_profile
              > client_transaction
              > client_wise_allocation
              > company
              > fundamental_report
              > generates
              > market_cap_wise
              > portfolio_creation
              > portfolio_holding
              > price
              > return_analytics
              > sector_wise
              > technical_strategy
            \v Trigger Functions

```

investment.sql*

On-Prem Project/postgres@PostgreSQL 16

Query Query History

```

24 -- 3. Fundamental_Report
25 DROP TABLE IF EXISTS Fundamental_Report CASCADE;
26 CREATE TABLE Fundamental_Report (
27   Report_ID SERIAL PRIMARY KEY,
28   ISIN VARCHAR(12) NOT NULL,
29   Date DATE NOT NULL,
30   PE_Ratio DECIMAL(10,2),
31   Debt DECIMAL(18,2),
32   Dividend DECIMAL(10,2),
33   Earnings_Per_Share DECIMAL(10,2),
34   FOREIGN KEY (ISIN) REFERENCES Company(ISIN)
35 );
36
37 Select * From Fundamental_Report;
38

```

Data Output Messages Notifications

	report_id [PK] integer	isin character varying(12)	date date	pe_ratio numeric(10,2)	debt numeric(18,2)	dividend numeric(10,2)	earnings_per_share numeric(10,2)
1	1	US0100000001	2024-10-03	54.24	58059.95	3.60	5.27
2	2	US0100000002	2024-10-03	38.69	94677.94	1.59	8.93
3	3	US0100000003	2024-10-03	21.65	100835.55	3.36	1.56
4	4	US0100000004	2024-10-03	54.14	87762.43	3.15	2.27
5	5	US0100000005	2024-10-03	24.53	55049.80	2.57	4.12
6	6	US0100000006	2024-10-03	19.06	100579.97	3.25	9.56
7	7	US0100000007	2024-10-03	58.39	25703.15	2.23	5.14
8	8	US0100000008	2024-10-03	21.18	57472.22	4.85	6.02
9	9	US0100000009	2024-10-03	35.02	19484.89	1.44	1.31
10	10	US0100000010	2024-10-03	18.10	70585.59	3.12	1.37
11	11	US0100000001	2024-10-02	37.20	10919.02	3.97	8.94
12	12	US0100000002	2024-10-02	39.88	11523.84	0.96	9.31
13	13	US0100000003	2024-10-02	54.00	96516.92	0.37	7.08
14	14	US0100000004	2024-10-02	59.24	47779.16	2.09	4.29
15	15	US0100000005	2024-10-02	38.39	75849.35	0.44	4.30
16	16	US0100000006	2024-10-02	31.13	75123.75	2.66	4.37
17	17	US0100000007	2024-10-02	58.43	15561.16	2.55	1.85
18	18	US0100000008	2024-10-02	45.18	15618.68	2.97	3.38
19	19	US0100000009	2024-10-02	40.16	53096.19	3.17	9.66

Total rows: 30 of 30 Query complete 00:00:00.260 Ln 37, Col 34

4. Technical_Strategy:

Object Explorer

```

    \v Servers (1)
      \v PostgreSQL 16
        \v Databases (3)
          > Northwind
          > On-Prem Project
            > Casts
            > Catalogs
            > Event Triggers
            > Extensions
            > Foreign Data Wrappers
            > Languages
            > Publications
          \v Schemas (2)
            \diamond Investment
              > Aggregates
              > Collations
              > Domains
              > FTS Configurations
              > FTS Dictionaries
              > FTS Parsers
              > FTS Templates
              > Foreign Tables
              > Functions
              > Materialized Views
              > Operators
              > Procedures
              > Sequences
            \v Tables (13)
              > client_profile
              > client_transaction
              > client_wise_allocation
              > company
              > fundamental_report
              > generates
              > market_cap_wise
              > portfolio_creation
              > portfolio_holding
              > price
              > return_analytics
              > sector_wise
              > technical_strategy
            \v Trigger Functions

```

investment.sql*

On-Prem Project/postgres@PostgreSQL 16

Query Query History

```

39 \v CREATE TABLE Technical_Strategy (
40   Strategy_ID SERIAL PRIMARY KEY,
41   Price_ID INTEGER NOT NULL,
42   Volatility DECIMAL(10,4),
43   Volume DECIMAL(18,2),
44   Simple_Moving_Average DECIMAL(18,2),
45   Exponential_Moving_Average DECIMAL(18,2),
46   MACD DECIMAL(10,4),
47   RSI DECIMAL(5,2),
48   Sharpe_Ratio DECIMAL(10,4),
49   FOREIGN KEY (Price_ID) REFERENCES Price(Price_ID)
50 );
51
52 Select * From Technical_Strategy;
53

```

Data Output Messages Notifications

	strategy_id [PK] integer	price_id integer	volatility numeric(10,4)	volume numeric(18,2)	simple_moving_average numeric(18,2)	exponential_moving_average numeric(18,2)	macd numeric(10,4)	rsi numeric(5,2)	sharpe_ratio numeric(10,4)
1	1	1	0.0569	3364695.86	1770.52	1779.55	0.0088	54.16	1.4710
2	2	2	0.0184	5058904.91	1712.63	1712.15	0.0605	42.74	1.3184
3	3	3	0.0196	5144177.40	1926.66	1923.08	0.0757	20.49	0.3207
4	4	4	0.0259	5323775.19	236.90	236.82	0.0138	27.03	0.9365
5	5	5	0.0467	2676692.32	1362.34	1355.08	0.0873	43.05	1.5363
6	6	6	0.0521	5932209.20	398.94	387.47	0.0282	28.03	0.8690
7	7	7	0.0168	2756547.80	642.02	651.56	0.0779	55.46	0.2655
8	8	8	0.0304	1431875.37	2012.43	2023.70	0.0667	37.49	1.4782
9	9	9	0.0382	5316517.35	957.57	966.13	0.0087	77.60	0.9734
10	10	10	0.0544	2689923.37	65.28	57.89	0.0275	71.57	1.4520
11	11	11	0.0286	5269261.89	1462.53	1470.48	0.0336	22.35	1.4579
12	12	12	0.0532	4323102.31	672.60	677.58	0.0458	63.20	1.1788
13	13	13	0.0450	5959860.64	790.80	796.52	0.0617	43.70	0.5138
14	14	14	0.0120	3765994.64	1691.11	1691.94	0.0271	45.98	1.5802
15	15	15	0.0260	5683662.19	253.61	244.39	0.0992	32.05	1.6614
16	16	16	0.0381	3788293.97	515.66	523.21	0.0821	58.11	0.9442
17	17	17	0.0272	4972831.48	1912.36	1896.12	0.0487	33.46	1.4419
18	18	18	0.0302	1870105.69	1444.11	1439.86	0.0105	74.59	1.4469
19	19	19	0.0550	4051521.78	1178.74	1178.48	0.0937	21.49	1.0033

Total rows: 100 of 100 Query complete 00:00:00.239 Ln 52, Col 1

5. Portfolio Creation

The screenshot shows the pgAdmin interface with the Object Explorer on the left and a query editor on the right.

Object Explorer:

- Servers (1) > PostgreSQL 16 > Databases (3) > On-Prem Project > investment
- Tables (13) > client_profile, client_transaction, client_wise_allocation, company, fundamental_report, generates, market_cap_wise, portfolio_creation, portfolio_holding, price, return_analytics, sector_wise, technical_strategy

Query Editor:

```

51 -- 5. Portfolio_Creation
52 DROP TABLE IF EXISTS Portfolio_Creation CASCADE;
53 CREATE TABLE Portfolio_Creation (
54     Portfolio_ID SERIAL PRIMARY KEY,
55     Strategy_ID INTEGER,
56     Report_ID INTEGER,
57     Date DATE NOT NULL,
58     FOREIGN KEY (Strategy_ID) REFERENCES Technical_Strategy(Strategy_ID),
59     FOREIGN KEY (Report_ID) REFERENCES Fundamental_Report(Report_ID)
60 );
61
62
63 Select * From Portfolio_Creation;
64

```

Data Output:

portfolio_id [PK] integer	strategy_id integer	report_id integer	date date
1	1	1	2024-10-03
2	2	2	2024-10-03
3	3	3	2024-10-03
4	4	4	2024-10-03
5	5	5	2024-10-03
6	6	6	2024-10-03
7	7	7	2024-10-03
8	8	8	2024-10-03
9	9	9	2024-10-03
10	10	10	2024-10-03
11	11	11	2024-10-03
12	12	12	2024-10-03
13	13	13	2024-10-03
14	14	14	2024-10-03
15	15	15	2024-10-03
16	16	16	2024-10-03
17	17	17	2024-10-03
18	18	18	2024-10-03
19	19	19	2024-10-03
20	20	20	2024-10-03

Total rows: 100 of 100 Query complete 00:00:00.293 Ln 63, Col 1

6. Portfolio_Holdings

The screenshot shows the pgAdmin interface with the Object Explorer on the left and a query editor on the right.

Object Explorer:

- Servers (1) > PostgreSQL 16 > Databases (3) > On-Prem Project > investment
- Tables (13) > client_profile, client_transaction, client_wise_allocation, company, fundamental_report, generates, market_cap_wise, portfolio_creation, portfolio_holding, price, return_analytics, sector_wise, technical_strategy

Query Editor:

```

62
63 -- 6. Portfolio_Holding
64 DROP TABLE IF EXISTS Portfolio_Holding CASCADE;
65 CREATE TABLE Portfolio_Holding (
66     Holding_ID SERIAL PRIMARY KEY,
67     Portfolio_ID INTEGER NOT NULL,
68     ISIN VARCHAR(12) NOT NULL,
69     Quantity INTEGER NOT NULL,
70     FOREIGN KEY (Portfolio_ID) REFERENCES Portfolio_Creation(Portfolio_ID),
71     FOREIGN KEY (ISIN) REFERENCES Company(ISIN)
72 );
73
74 Select * From Portfolio_Holding;
75

```

Data Output:

holding_id [PK] integer	portfolio_id integer	isn character varying (12)	quantity integer
1	1	US0100000001	34
2	2	US0100000002	54
3	3	US0100000003	26
4	4	US0100000004	52
5	5	US0100000005	95
6	6	US0100000006	70
7	7	US0100000007	3
8	8	US0100000008	83
9	9	US0100000009	29
10	10	US0100000010	94
11	11	US1000000001	41
12	12	US1000000002	41
13	13	US1000000003	71
14	14	US1000000004	44
15	15	US1000000005	13
16	16	US1000000006	14
17	17	US1000000007	92
18	18	US1000000008	89
19	19	US1000000009	84
20	20	US1000000010	91

Total rows: 500 of 500 Query complete 00:00:00.639 Ln 74, Col 33

7. Client_Profile:

Object Explorer

```

    \-\ Servers (1)
        \-\ PostgreSQL 16
            \-\ Databases (3)
                > \-\ Northwind
                > \-\ On-Prem Project
                    > \-\ Casts
                    > \-\ Catalogs
                    > \-\ Event Triggers
                    > \-\ Extensions
                    > \-\ Foreign Data Wrappers
                    > \-\ Languages
                    > \-\ Publications
                    > \-\ Schemas (2)
                        > \-\ investment
                            > \-\ Aggregates
                            > \-\ Collations
                            > \-\ Domains
                            > \-\ FTS Configurations
                            > \-\ FTS Dictionaries
                            > \-\ FTS Parsers
                            > \-\ FTS Templates
                            > \-\ Foreign Tables
                            > \-\ Functions
                            > \-\ Materialized Views
                            > \-\ Operators
                            > \-\ Procedures
                            > \-\ Sequences
                            > \-\ Tables (13)
                                > \-\ client_profile
                                > \-\ client_transaction
                                > \-\ client_wise_allocation
                                > \-\ company
                                > \-\ fundamental_report
                                > \-\ generates
                                > \-\ market_cap_wise
                                > \-\ portfolio_creation
                                > \-\ portfolio_holding
                                > \-\ price
                                > \-\ return_analytics
                                > \-\ sector_wise
                                > \-\ technical_strategy
                                > \-\ Trigger Functions

```

investment.sql*

```

-- 7. Client_Profile
DROP TABLE IF EXISTS Client_Profile CASCADE;
CREATE TABLE Client_Profile (
    Client_ID VARCHAR(20) PRIMARY KEY NOT NULL,
    Client_Name VARCHAR(100),
    Expected_Return DECIMAL(5,2),
    Investment_Amount DECIMAL(18,2),
    Investment_Duration INTEGER,
    Risk_Appetite VARCHAR(20)
);
Select * From Client_Profile;

```

Data Output

client_id	client_name	expected_return	investment_amount	investment_duration	risk_appetite
4e14532d-8e1c-4758-b	Client 1	6.45	198135.05	8	Medium
a00ddab6-5e1a-494b-9	Client 2	11.45	119655.92	5	Low
7538378be-d061-4401-a	Client 3	6.60	74405.38	4	High
ace52516-59da-4900-9	Client 4	9.97	142789.06	4	Medium
be151295-4062-4276-a	Client 5	10.28	52160.67	2	Low
d3d4925-702e-4663-a	Client 6	9.93	191483.39	8	High
2a4fcfcf-6275-4fe9-b	Client 7	5.88	110511.45	2	Medium
805b98db-8eb4-4b6b-9	Client 8	13.17	148566.44	11	Low
64ba3ef5-c18d-4bad-8	Client 9	9.03	169071.09	9	High
b0dff5ed-00af-4a9c-a	Client 10	11.87	127039.38	9	Medium
f05e4bc9-dc2c-4887-a	Client 11	9.60	59236.50	3	Low
304425ec-d305-48c9-b	Client 12	11.21	164074.76	3	High
5da363d6-793d-447d-9	Client 13	8.64	77252.58	9	Medium
899db1f0-c70e-4304-a	Client 14	13.39	135811.91	9	Low
acb288e-78ce-4baa-8	Client 15	12.04	70844.95	9	High
9c7f53a1-4b5a-43a6-b	Client 16	14.58	130669.81	9	Medium
6abb5bd2-15e3-4aa5-b	Client 17	13.29	128692.85	7	Low
13bb93f7-299d-4a90-a	Client 18	10.90	52173.38	8	High
d40b5962-e345-4f88-b	Client 19	7.88	64001.72	8	Medium
60cef745-14d9-4302-a	Client 20	11.27	94100.99	6	Low

Total rows: 50 of 50 Query complete 00:00:00.229 Ln 85, Col 1

8. Generates:

Object Explorer

```

    \-\ Servers (1)
        \-\ PostgreSQL 16
            \-\ Databases (3)
                > \-\ Northwind
                > \-\ On-Prem Project
                    > \-\ Casts
                    > \-\ Catalogs
                    > \-\ Event Triggers
                    > \-\ Extensions
                    > \-\ Foreign Data Wrappers
                    > \-\ Languages
                    > \-\ Publications
                    > \-\ Schemas (2)
                        > \-\ investment
                            > \-\ Aggregates
                            > \-\ Collations
                            > \-\ Domains
                            > \-\ FTS Configurations
                            > \-\ FTS Dictionaries
                            > \-\ FTS Parsers
                            > \-\ FTS Templates
                            > \-\ Foreign Tables
                            > \-\ Functions
                            > \-\ Materialized Views
                            > \-\ Operators
                            > \-\ Procedures
                            > \-\ Sequences
                            > \-\ Tables (13)
                                > \-\ client_profile
                                > \-\ client_transaction
                                > \-\ client_wise_allocation
                                > \-\ company
                                > \-\ fundamental_report
                                > \-\ generates
                                > \-\ market_cap_wise
                                > \-\ portfolio_creation
                                > \-\ portfolio_holding
                                > \-\ price
                                > \-\ return_analytics
                                > \-\ sector_wise
                                > \-\ technical_strategy
                                > \-\ Trigger Functions

```

investment.sql*

```

-- 8. Generates
DROP TABLE IF EXISTS Generates CASCADE;
CREATE TABLE Generates (
    Generation_ID SERIAL PRIMARY KEY,
    Client_ID VARCHAR(20) NOT NULL,
    Portfolio_ID INTEGER NOT NULL,
    Generation_Date DATE NOT NULL,
    FOREIGN KEY (Client_ID) REFERENCES Client_Profile(Client_ID),
    FOREIGN KEY (Portfolio_ID) REFERENCES Portfolio_Creation(Portfolio_ID)
);
Select * From Generates;

```

Data Output

generation_id	client_id	portfolio_id	generation_date
1	4e14532d-8e1c-4758-b	1	2024-10-04
2	4e14532d-8e1c-4758-b	2	2024-09-16
3	4e14532d-8e1c-4758-b	3	2024-09-29
4	4e14532d-8e1c-4758-b	4	2024-09-15
5	4e14532d-8e1c-4758-b	5	2024-09-18
6	4e14532d-8e1c-4758-b	6	2024-09-19
7	4e14532d-8e1c-4758-b	7	2024-09-21
8	4e14532d-8e1c-4758-b	8	2024-09-14
9	4e14532d-8e1c-4758-b	9	2024-10-03
10	4e14532d-8e1c-4758-b	10	2024-09-08
11	4e14532d-8e1c-4758-b	11	2024-09-17
12	4e14532d-8e1c-4758-b	12	2024-09-17
13	4e14532d-8e1c-4758-b	13	2024-09-21
14	4e14532d-8e1c-4758-b	14	2024-09-17
15	4e14532d-8e1c-4758-b	15	2024-10-02
16	4e14532d-8e1c-4758-b	16	2024-09-26
17	4e14532d-8e1c-4758-b	17	2024-09-29
18	4e14532d-8e1c-4758-b	18	2024-09-08
19	4e14532d-8e1c-4758-b	19	2024-10-02
20	4e14532d-8e1c-4758-b	20	2024-09-21

Total rows: 200 of 200 Query complete 00:00:00.182 Ln 96, Col 1

9. Client_Transaction:

The screenshot shows the Object Explorer pane with the following structure:

- Servers (1)
 - PostgreSQL 16
 - Databases (3)
 - Northwind
 - On-Prem Project
 - investment

The investment database is selected. The Tables section contains 13 tables, including client_profile, client_transaction, client_wise_allocation, company, fundamental_report, generates, market_cap_wise, portfolio_creation, portfolio_holding, price, return_analytics, sector_wise, and technical_strategy.

The Query Editor window displays the following SQL code and its execution results:

```

-- 9. Client_Transaction
DROP TABLE IF EXISTS Client_Transaction CASCADE;
CREATE TABLE Client_Transaction (
    Transaction_ID SERIAL PRIMARY KEY,
    Client_ID VARCHAR(20) NOT NULL,
    Date DATE NOT NULL,
    Amount DECIMAL(18,2) NOT NULL,
    FOREIGN KEY (Client_ID) REFERENCES client_Profile(Client_ID)
);
Select * From Client_Transaction;

```

Data Output

transaction_id	client_id	date	amount
1	4e14532d-8e1c-4758-b	2024-09-19	16103.71
2	a00dd46b-5e1a-494b-9	2024-09-28	9039.44
3	7538378e-d061-44b1-a	2024-09-06	22513.32
4	ace52516-59da-4900-9	2024-09-07	25166.00
5	b1e151295-4062-4276-a	2024-09-28	49392.47
6	d3c4d925-702e-4663-a	2024-09-30	34199.61
7	2a44cef6-6275-4fe9-a	2024-09-25	23029.59
8	805b98db-8eb4-4b68-9	2024-09-26	8124.60
9	648a3ef5-c18d-4bad-8	2024-09-04	17752.92
10	b0df15ed-00af-4a9c-9	2024-09-29	24145.42
11	f05e4bc9-dc2c-4887-a	2024-09-24	32291.67
12	304425ec-d305-48c9-b	2024-09-04	22967.76
13	5da363d6-793d-447d-9	2024-09-26	40211.11
14	89bd8f10-c70e-4304-a	2024-09-12	48197.14
15	acbb298e-78ce-4baa-8	2024-09-14	39259.45
16	9c7f53a1-4b5a-43a6-b	2024-10-01	43057.13
17	6abb85d2-15e3-4aa5-b	2024-09-18	46256.52
18	138b93f7-299d-4a99-a	2024-09-09	10358.93
19	d40b5962-e345-4f88-b	2024-09-29	8786.53
20	60cccf745-14d9-4302-a	2024-09-14	34868.23

Total rows: 250 of 250 Query complete 00:00:00.159 Ln 106, Col 1

10. Client_Wise_Allocation

The screenshot shows the Object Explorer pane with the following structure:

- Servers (1)
 - PostgreSQL 16
 - Databases (3)
 - Northwind
 - On-Prem Project
 - investment

The investment database is selected. The Tables section contains 13 tables, including client_profile, client_transaction, client_wise_allocation, company, fundamental_report, generates, market_cap_wise, portfolio_creation, portfolio_holding, price, return_analytics, sector_wise, and technical_strategy.

The Query Editor window displays the following SQL code and its execution results:

```

-- 10. Client_Wise_Allocation
DROP TABLE IF EXISTS Client_Wise_Allocation CASCADE;
CREATE TABLE Client_Wise_Allocation (
    Allocation_ID SERIAL PRIMARY KEY,
    Client_ID VARCHAR(20) NOT NULL,
    Portfolio_ID INTEGER NOT NULL,
    Date DATE NOT NULL,
    Investment_Amount DECIMAL(18,2),
    Current_Value DECIMAL(18,2),
    Percentage_Gain_Loss DECIMAL(5,2),
    FOREIGN KEY (Client_ID) REFERENCES Client_Profile(Client_ID),
    FOREIGN KEY (Portfolio_ID) REFERENCES Portfolio_Creation(Portfolio_ID)
);
Select * From Client Wise Allocation;

```

Data Output

allocation_id	client_id	portfolio_id	date	investment_amount	current_value	percentage_gain_loss
1	4e14532d-8e1c-4758-b	1	2024-09-27	47009.37	21269.77	4.04
2	4e14532d-8e1c-4758-b	2	2024-09-16	32992.28	13534.81	2.69
3	4e14532d-8e1c-4758-b	3	2024-09-15	36868.46	57220.48	-1.75
4	4e14532d-8e1c-4758-b	4	2024-09-13	35146.79	16853.94	6.44
5	4e14532d-8e1c-4758-b	5	2024-09-12	20679.50	41519.77	4.85
6	4e14532d-8e1c-4758-b	6	2024-09-06	40057.96	63099.99	7.02
7	4e14532d-8e1c-4758-b	7	2024-09-09	21880.25	13030.72	-3.33
8	4e14532d-8e1c-4758-b	8	2024-09-20	53843.38	33630.90	7.24
9	4e14532d-8e1c-4758-b	9	2024-09-13	31745.70	37050.47	5.78
10	4e14532d-8e1c-4758-b	10	2024-09-30	43880.77	29237.11	3.31
11	4e14532d-8e1c-4758-b	11	2024-10-04	17017.82	22112.91	-3.90
12	4e14532d-8e1c-4758-b	12	2024-09-20	9559.80	9205.32	-3.34
13	4e14532d-8e1c-4758-b	13	2024-09-22	34824.71	45112.43	-4.44
14	4e14532d-8e1c-4758-b	14	2024-09-27	50483.68	46132.64	0.93
15	4e14532d-8e1c-4758-b	15	2024-09-26	51173.83	63294.17	4.36
16	4e14532d-8e1c-4758-b	16	2024-10-04	18577.55	44426.43	-2.63
17	4e14532d-8e1c-4758-b	17	2024-09-15	41796.38	42572.43	5.08
18	4e14532d-8e1c-4758-b	18	2024-10-01	7323.46	22584.65	-5.61
19	4e14532d-8e1c-4758-b	19	2024-09-19	22025.98	47640.29	8.49
20	4e14532d-8e1c-4758-b	20	2024-09-06	11714.99	11596.13	-3.88

Total rows: 200 of 200 Query complete 00:00:00.178 Ln 121, Col 1

11. Return_Analytics:

Object Explorer

```

    \-\-\ 11. Return_Analytics
    DROP TABLE IF EXISTS Return_Analytics CASCADE;
    CREATE TABLE Return_Analytics (
        Return_ID SERIAL PRIMARY KEY,
        Portfolio_ID INTEGER NOT NULL,
        Date DATE NOT NULL,
        Benchmark_Return DECIMAL(5,2),
        Portfolio_Return DECIMAL(5,2),
        FOREIGN KEY (Portfolio_ID) REFERENCES Portfolio_Creation(Portfolio_ID)
    );
    Select * From Return_Analytics;

```

Data Output

	return_id [PK] integer	portfolio_id integer	date date	benchmark_return numeric (5,2)	portfolio_return numeric (5,2)
1	1	1	2024-09-21	12.81	7.12
2	2	2	2024-09-11	6.51	10.02
3	3	3	2024-09-30	7.91	7.02
4	4	4	2024-09-16	14.87	9.18
5	5	5	2024-10-01	10.62	11.10
6	6	6	2024-09-26	7.49	5.88
7	7	7	2024-09-24	8.94	12.81
8	8	8	2024-09-22	12.98	6.15
9	9	9	2024-09-07	13.00	9.36
10	10	10	2024-09-16	6.11	5.58
11	11	11	2024-09-18	11.83	12.23
12	12	12	2024-09-05	7.07	13.86
13	13	13	2024-09-26	10.20	14.00
14	14	14	2024-09-05	5.77	7.74
15	15	15	2024-09-21	7.34	11.42
16	16	16	2024-09-11	10.38	14.44
17	17	17	2024-09-15	13.78	6.43
18	18	18	2024-09-21	11.46	5.45
19	19	19	2024-09-10	5.38	11.19
20	20	20	2024-09-14	6.03	11.66

Total rows: 100 of 100 Query complete 00:00:00.203 Ln 133, Col 15

12. Sector_Wise:

Object Explorer

```

    -- 12. Sector_Wise
    DROP TABLE IF EXISTS Sector_Wise CASCADE;
    CREATE TABLE Sector_Wise (
        Sector_Return_ID SERIAL PRIMARY KEY,
        Sector_ID INTEGER NOT NULL,
        Sector VARCHAR(50) NOT NULL,
        Benchmark_Return DECIMAL(5,2),
        Portfolio_Return DECIMAL(5,2),
        FOREIGN KEY (Sector_ID) REFERENCES Return_Analytics(Return_ID)
    );
    -- Note: Foreign key on Sector is not enforced due to potential non-uniqueness in Company(Sector)
    Select * From Sector_Wise;

```

Data Output

	sector_return_id [PK] integer	return_id integer	sector character varying (50)	benchmark_return numeric (5,2)	portfolio_return numeric (5,2)
1	1	2	Industrials	12.73	14.39
2	2	8	Information Technology	9.29	6.00
3	3	8	Technology	13.78	10.41
4	4	6	Financials	7.13	6.12
5	5	2	Consumer Staples	5.53	12.13
6	6	1	Information Technology	10.16	8.50
7	7	3	Financials	5.43	6.04
8	8	1	Financials	9.29	12.89
9	9	4	Communication Services	7.32	10.04
10	10	5	Financials	6.24	13.65
11	11	9	Communication Services	7.76	13.67
12	12	6	Industrials	7.73	10.70
13	13	6	Consumer Staples	12.90	11.82
14	14	6	Materials	12.01	6.41
15	15	1	Healthcare	10.18	8.33
16	16	7	Healthcare	14.18	5.93
17	17	9	Information Technology	12.51	8.00
18	18	7	Information Technology	12.47	10.01
19	19	6	Information Technology	9.60	9.32
20	20	2	Information Technology	7.90	5.83

Total rows: 104 of 104 Query complete 00:00:00.381 Ln 149, Col 1

13. Market_Cap_Wise:

The screenshot shows the pgAdmin interface with the following details:

- Object Explorer:** Shows the database structure under "PostgreSQL 16" and "Northwind".
- Query Editor:** The current file is "investment.sql" with the following content:

```
151 -- 13. Market_Cap_Wise
152 DROP TABLE IF EXISTS Market_Cap_Wise CASCADE;
153 CREATE TABLE Market_Cap_Wise (
154     MarketCap_Return_ID SERIAL PRIMARY KEY,
155     Return_ID INTEGER NOT NULL,
156     Market_Cap_Category VARCHAR(20) NOT NULL,
157     Benchmark_Return DECIMAL(5,2),
158     Portfolio_Return DECIMAL(5,2),
159     FOREIGN KEY (Return_ID) REFERENCES Return_Analytics(Return_ID)
160     -- Note: Foreign key on Market_Cap_Category is not enforced due to potential non-uniqueness
161 );
162
163 Select * From Market_Cap_Wise;
164
```
- Data Output:** A table showing the data from the Market_Cap_Wise table.

	marketcap_return_id [PK] integer	return_id integer	market_cap_category character varying(20)	benchmark_return numeric(5,2)	portfolio_return numeric(5,2)
1		1	9 Mid Cap	7.93	11.02
2		2	107 Mid Cap	10.29	13.07
3		3	109 Large Cap	7.05	13.18
4		4	1 Mid Cap	13.68	6.83
5		5	5 Mid Cap	13.19	12.91
6		6	102 Mega Cap	6.47	14.01
7		7	4 Mega Cap	12.99	8.67
8		8	109 Mega Cap	9.96	8.61
9		9	2 Large Cap	8.10	14.03
10		10	5 Mega Cap	14.75	12.06
11		11	7 Mid Cap	6.44	13.24
12		12	102 Mid Cap	5.97	7.34
13		13	1 Mega Cap	10.51	9.54
14		14	104 Mega Cap	7.28	12.26
15		15	6 Mid Cap	7.40	11.92
16		16	104 Mid Cap	14.68	6.81
17		17	106 Large Cap	6.08	13.45
18		18	108 Mega Cap	9.71	8.69
19		19	6 Large Cap	8.07	12.91
20		20	101 Mega Cap	14.48	10.46

Total rows: 54 of 54 Query complete 00:00:00.165 Ln 163, Col 1