2. EigenSim is a web-based algorithmic trading simulator that allow users to backtest custom or predefined trading strategies on historical market data. Users can configure strategies using a set of parameters and evaluate their performance through generated backtest reports
3. Attach later
4.

- Users(user_id: VARCHAR2(12), first_name: VARCHAR2(20), last_name: VARCHAR2(20), password_hash: VARCHAR2(100)
  - PK: user_id
  - Constraints: password_hash NOT NULL
- Strategy(strategy_id: VARCHAR2(12), strategy_type: VARCHAR2(20))
  - PK: strategy_id
- CustomStrategy(strategy_id: CHAR(12), name: VARCHAR2(50), description: CLOB, created_date: DATE, custom_prompt: CLOB)
  - PK: strategy_id
  - FK: strategy_id -> Strategy(strategy_id)
- PredefinedStrategy(strategy_id: CHAR(12), name: VARCHAR2(50), description: CLOB, created_date: DATE, logic: CLOB)
  - PK: strategy_id
  - FK: strategy_id -> Strategy(strategy_id)
- Parameter(parameter_id: CHAR(12), name: VARCHAR2(30), type: VARCHAR2(10))
  - PK: parameter_id
- ParameterValue(**strategy_id**: CHAR(12), **parameter_id**: CHAR(12), assigned_value: VARCHAR2(100))
  - PK: (strategy_id, parameter_id)
  - FK: strategy_id → Strategy(strategy_id), parameter_id → Parameter(parameter_id)
- Ticker(ticker_symbol: VARCHAR2(10), company_name: VARCHAR2(100), exchange: VARCHAR2(20))
  - PK: ticker_symbol
- Backtest(backtest_id: CHAR(12), **strategy_id**: CHAR(12), **user_id**: VARCHAR2(12), **ticker_symbol**: VARCHAR2(10), start_date: DATE, end_date: DATE, result_summary: CLOB)
  - PK: backtest_id
  - FK: strategy_id -> Strategy(strategy_id), user_id -> Users(user_id), ticker_symbol -> Ticker(ticker_symbol)
- Report(report_id: VARCHAR2(20), **backtest_id**: VARCHAR2(12), generated_at: TIMESTAMP, total_return: NUMBER, annualized_return: NUMBER, sharpe_ratio: NUMBER, max_drawdown: NUMBER, win_rate: NUMBER, trade_count: NUMBER, t_stat: NUMBER, p_value: NUMBER, confidence_95_low: NUMBER, confidence_95_high: NUMBER)
  - PK: report_id
  - FK: backtest_id -> Backtest(backtest_id)
  - Constraint: backtest_id is UNIQUE

- Trade(trade_id: VARCHAR2(12), **backtest_id**: VARCHAR2(12), trade_time: TIMESTAMP, price: NUMBER, quantity: NUMBER, price: side VARCHAR2(4))
  - PK: trade_id
  - FK: backtest_id -> Backtest(backtest_id)

5.

- Users
  - user_id → first_name, last_name, password_hash (PK FD)
- Strategy
  - strategy_id → strategy_type (PK FD)
- CustomStrategy
  - strategy_id → name, description, created_date, custom_prompt (PK FD)
- PredefinedStrategy
  - strategy_id → name, description, created_date, logic (PK FD)
- Parameter
  - parameter_id → name, type (PK FD)
  - name → type
- ParameterValue
  - (strategy_id, parameter_id) → assigned_value (PK FD)
- Ticker
  - ticker_symbol → company_name, exchange (PK FD)
- Backtest
  - backtest_id → strategy_id, user_id, ticker_symbol, start_date, end_date, result_summary (PK FD)
- Report
  - report_id → backtest_id, generated_at, total_return, annualized_return, sharpe_ratio, max_drawdown, win_rate, trade_count, t_stat, p_value, confidence_95_low, confidence_95_high (PK FD)
  - backtest_id → report_id (Non-key FD — 1:1 enforced by UNIQUE constraint)
- Trade
  - trade_id → backtest_id, trade_time, price, quantity, side (PK FD)

6.

Everything is already in BCNF except for Parameters and Report
Parameter: parameter_id → name, type, name → type

Normalize into 2 table: Parameter(parameter_id, name), ParameterType(name, type)

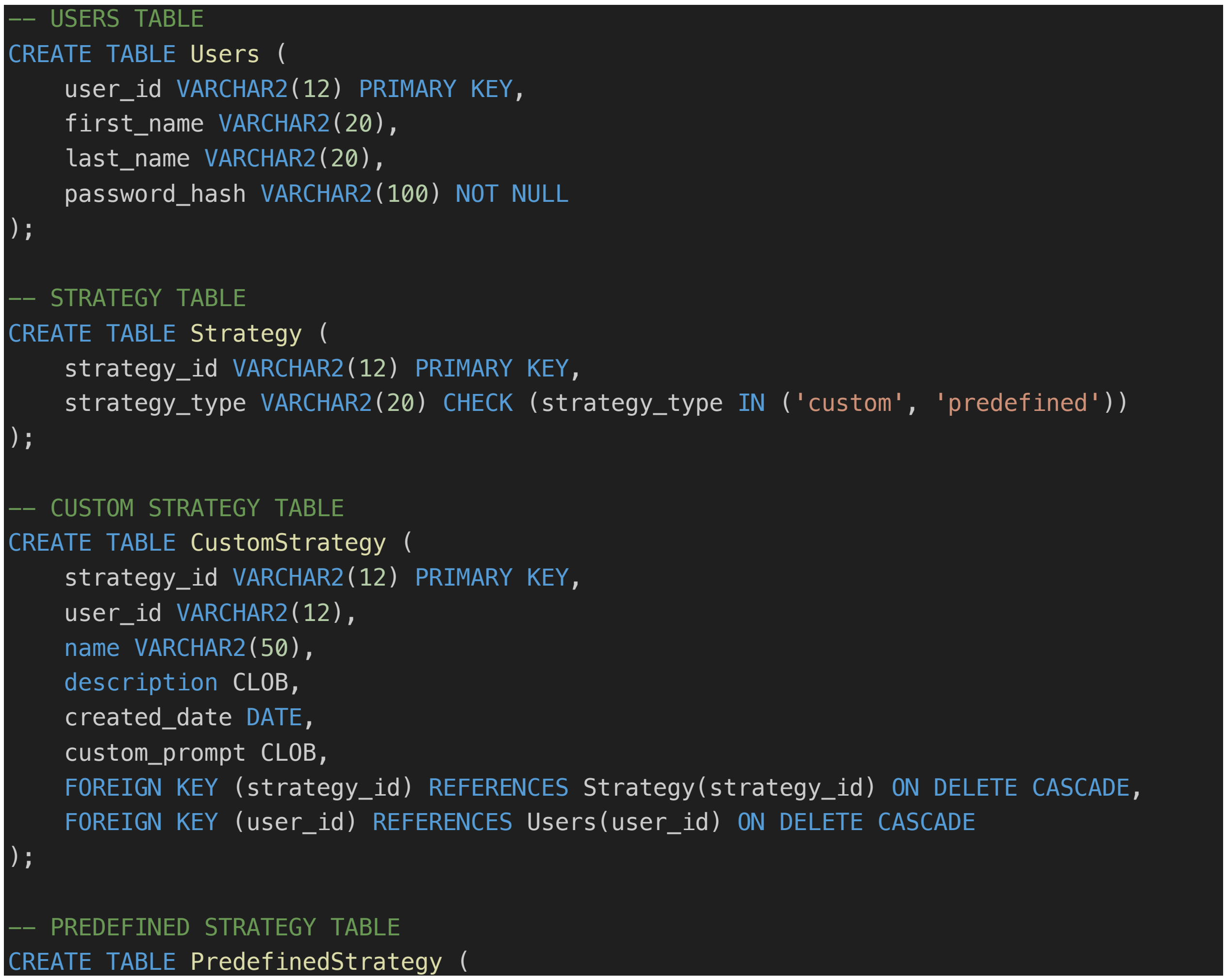
Report: report_id → all, backtest_id → report_id

Normalize into 2 table: ReportMetadata(report_id, backtest_id), ReportStats (report_id, generated_at, total_return, annualized_return, sharpe_ratio, max_drawdown, win_rate, trade_count, t_stat, p_value, confidence_95_low, confidence_95_high)

Table Name	PK	FKs
Users	user_id	
Strategy	strategy_id	
CustomStrategy	strategy_id	strategy_id -> Strategy

PredefinedStrategy	strategy_id	strategy_id -> Strategy
Parameter	parameter_id	
ParameterType	name	
ParameterValue	(strategy_id, parameter_id)	strategy_id -> Strategy, parameter_id -> Parameter
Ticker	ticker_symbol	
Backtest	backtest_id	strategy_id, user_id, ticker_symbol
ReportMetadata	report_id	backtest_id
ReportStats	report_id	report_id → ReportMetadata
Trade	trade_id	backtest_id

7. From vscode for better format



```
-- USERS TABLE
CREATE TABLE Users (
    user_id VARCHAR2(12) PRIMARY KEY,
    first_name VARCHAR2(20),
    last_name VARCHAR2(20),
    password_hash VARCHAR2(100) NOT NULL
);

-- STRATEGY TABLE
CREATE TABLE Strategy (
    strategy_id VARCHAR2(12) PRIMARY KEY,
    strategy_type VARCHAR2(20) CHECK (strategy_type IN ('custom', 'predefined'))
);

-- CUSTOM STRATEGY TABLE
CREATE TABLE CustomStrategy (
    strategy_id VARCHAR2(12) PRIMARY KEY,
    user_id VARCHAR2(12),
    name VARCHAR2(50),
    description CLOB,
    created_date DATE,
    custom_prompt CLOB,
    FOREIGN KEY (strategy_id) REFERENCES Strategy(strategy_id) ON DELETE CASCADE,
    FOREIGN KEY (user_id) REFERENCES Users(user_id) ON DELETE CASCADE
);

-- PREDEFINED STRATEGY TABLE
CREATE TABLE PredefinedStrategy (
```

```sql
    strategy_id VARCHAR2(12) PRIMARY KEY,
    name VARCHAR2(50),
    description CLOB,
    created_date DATE,
    logic CLOB,
    FOREIGN KEY (strategy_id) REFERENCES Strategy(strategy_id) ON DELETE CASCADE
);

-- PARAMETER TABLE
CREATE TABLE Parameter (
    parameter_id VARCHAR2(12) PRIMARY KEY,
    name VARCHAR2(30) UNIQUE
);

-- PARAMETER TYPE TABLE (from normalization of Parameter)
CREATE TABLE ParameterType (
    name VARCHAR2(30) PRIMARY KEY,
    type VARCHAR2(10) CHECK (type IN ('int', 'float', 'bool', 'string'))
);

-- PARAMETER VALUE TABLE
CREATE TABLE ParameterValue (
    strategy_id VARCHAR2(12),
    parameter_id VARCHAR2(12),
    assigned_value VARCHAR2(100),
    PRIMARY KEY (strategy_id, parameter_id),
    FOREIGN KEY (strategy_id) REFERENCES Strategy(strategy_id) ON DELETE CASCADE,
    FOREIGN KEY (parameter_id) REFERENCES Parameter(parameter_id) ON DELETE CASCADE
);

-- TICKER TABLE
CREATE TABLE Ticker (
    ticker_symbol VARCHAR2(10) PRIMARY KEY,
    company_name VARCHAR2(100),
    exchange VARCHAR2(20)
);

-- BACKTEST TABLE
CREATE TABLE Backtest (
    backtest_id VARCHAR2(12) PRIMARY KEY,
    strategy_id VARCHAR2(12),
    user_id VARCHAR2(12),
    ticker_symbol VARCHAR2(10),
    start_date DATE,
    end_date DATE,
    result_summary CLOB,
    FOREIGN KEY (strategy_id) REFERENCES Strategy(strategy_id),
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (ticker_symbol) REFERENCES Ticker(ticker_symbol)
);

-- REPORT METADATA TABLE (from normalization of Report)
CREATE TABLE ReportMetadata (
    report_id VARCHAR2(20) PRIMARY KEY,
    backtest_id VARCHAR2(12) UNIQUE,
    FOREIGN KEY (backtest_id) REFERENCES Backtest(backtest_id) ON DELETE CASCADE
);

-- REPORT STATS TABLE
CREATE TABLE ReportStats (
    report_id VARCHAR2(20) PRIMARY KEY,
    generated_at TIMESTAMP,
    total_return NUMBER,
    annualized_return NUMBER,
    sharpe_ratio NUMBER,
    max_drawdown NUMBER,
    win_rate NUMBER,
    trade_count NUMBER,
    t_stat NUMBER,
    p_value NUMBER,
    confidence_95_low NUMBER,
    confidence_95_high NUMBER,
    FOREIGN KEY (report_id) REFERENCES ReportMetadata(report_id) ON DELETE CASCADE
);

-- TRADE TABLE
CREATE TABLE Trade (
    trade_id VARCHAR2(12) PRIMARY KEY,
    backtest_id VARCHAR2(12),
    trade_time TIMESTAMP,
    price NUMBER,
    quantity NUMBER,
    side VARCHAR2(4) CHECK (side IN ('BUY', 'SELL')),
    FOREIGN KEY (backtest_id) REFERENCES Backtest(backtest_id) ON DELETE CASCADE
);
```

8. again from vscode for better format. We register users on the website so no need to insert to Users table

```sql
-- Strategy
INSERT INTO Strategy VALUES ('S001', 'custom');
INSERT INTO Strategy VALUES ('S002', 'custom');
INSERT INTO Strategy VALUES ('S003', 'custom');
INSERT INTO Strategy VALUES ('S004', 'predefined');
INSERT INTO Strategy VALUES ('S005', 'predefined');

-- CustomStrategy
INSERT INTO CustomStrategy VALUES ('S001', 'tomcsvan', 'Custom 1', 'Desc 1', DATE '2024-01-01', 'Prompt 1');
INSERT INTO CustomStrategy VALUES ('S002', 'olivia', 'Custom 2', 'Desc 2', DATE '2024-01-01', 'Prompt 2');
INSERT INTO CustomStrategy VALUES ('S003', '111test', 'Custom 3', 'Desc 3', DATE '2024-01-01', 'Prompt 3');

-- PredefinedStrategy
INSERT INTO PredefinedStrategy VALUES ('S004', 'Predefined 4', 'Desc 4', DATE '2024-01-01', 'Logic 4');
INSERT INTO PredefinedStrategy VALUES ('S005', 'Predefined 5', 'Desc 5', DATE '2024-01-01', 'Logic 5');

-- Parameter
INSERT INTO Parameter VALUES ('P001', 'Param1');
INSERT INTO Parameter VALUES ('P002', 'Param2');
INSERT INTO Parameter VALUES ('P003', 'Param3');
INSERT INTO Parameter VALUES ('P004', 'Param4');
INSERT INTO Parameter VALUES ('P005', 'Param5');

-- ParameterType
INSERT INTO ParameterType VALUES ('Param1', 'float');
INSERT INTO ParameterType VALUES ('Param2', 'bool');
INSERT INTO ParameterType VALUES ('Param3', 'string');
INSERT INTO ParameterType VALUES ('Param4', 'int');
INSERT INTO ParameterType VALUES ('Param5', 'float');

-- ParameterValue
INSERT INTO ParameterValue VALUES ('S001', 'P001', 'val1');
INSERT INTO ParameterValue VALUES ('S001', 'P002', 'val2');
INSERT INTO ParameterValue VALUES ('S001', 'P003', 'val3');
INSERT INTO ParameterValue VALUES ('S001', 'P004', 'val4');
INSERT INTO ParameterValue VALUES ('S001', 'P005', 'val5');

-- Ticker
INSERT INTO Ticker VALUES ('AAPL', 'Company 1', 'NASDAQ');
INSERT INTO Ticker VALUES ('GOOGL', 'Company 2', 'NASDAQ');
INSERT INTO Ticker VALUES ('TSLA', 'Company 3', 'NASDAQ');
INSERT INTO Ticker VALUES ('MSFT', 'Company 4', 'NASDAQ');
INSERT INTO Ticker VALUES ('AMZN', 'Company 5', 'NASDAQ');

-- Backtest
INSERT INTO Backtest VALUES ('B001', 'S001', 'tomcsvan', 'AAPL', DATE '2022-01-01', DATE '2022-12-31', 'Summary 1');
INSERT INTO Backtest VALUES ('B002', 'S001', 'olivia', 'AAPL', DATE '2022-01-01', DATE
'2022-12-31', 'Summary 2');
INSERT INTO Backtest VALUES ('B003', 'S001', '111test', 'AAPL', DATE '2022-01-01',
DATE '2022-12-31', 'Summary 3');
INSERT INTO Backtest VALUES ('B004', 'S001', 'yanhe', 'AAPL', DATE '2022-01-01', DATE
'2022-12-31', 'Summary 4');
INSERT INTO Backtest VALUES ('B005', 'S001', 'tomcsvan', 'AAPL', DATE '2022-01-01',
DATE '2022-12-31', 'Summary 5');

-- ReportMetadata
INSERT INTO ReportMetadata VALUES ('R001', 'B001');
INSERT INTO ReportMetadata VALUES ('R002', 'B002');
INSERT INTO ReportMetadata VALUES ('R003', 'B003');
INSERT INTO ReportMetadata VALUES ('R004', 'B004');
INSERT INTO ReportMetadata VALUES ('R005', 'B005');

-- ReportStats
INSERT INTO ReportStats VALUES ('R001', TO_TIMESTAMP('2023-01-01 10:00:00', 'YYYY-MM-
DD HH24:MI:SS'), 0.10, 0.08, 1.1, -0.05, 0.55, 11, 2.5, 0.05, 0.05, 0.10);
INSERT INTO ReportStats VALUES ('R002', TO_TIMESTAMP('2023-01-02 10:00:00', 'YYYY-MM-
DD HH24:MI:SS'), 0.20, 0.16, 1.2, -0.10, 0.60, 12, 2.6, 0.05, 0.10, 0.20);
INSERT INTO ReportStats VALUES ('R003', TO_TIMESTAMP('2023-01-03 10:00:00', 'YYYY-MM-
DD HH24:MI:SS'), 0.30, 0.24, 1.3, -0.15, 0.65, 13, 2.7, 0.05, 0.15, 0.30);
INSERT INTO ReportStats VALUES ('R004', TO_TIMESTAMP('2023-01-04 10:00:00', 'YYYY-MM-
DD HH24:MI:SS'), 0.40, 0.32, 1.4, -0.20, 0.70, 14, 2.8, 0.05, 0.20, 0.40);
INSERT INTO ReportStats VALUES ('R005', TO_TIMESTAMP('2023-01-05 10:00:00', 'YYYY-MM-
DD HH24:MI:SS'), 0.50, 0.40, 1.5, -0.25, 0.75, 15, 2.9, 0.05, 0.25, 0.50);

-- Trade
INSERT INTO Trade VALUES ('T001', 'B001', TO_TIMESTAMP('2022-01-11 10:00:00', 'YYYY-
MM-DD HH24:MI:SS'), 101, 10, 'SELL');
INSERT INTO Trade VALUES ('T002', 'B001', TO_TIMESTAMP('2022-01-12 10:00:00', 'YYYY-
MM-DD HH24:MI:SS'), 102, 20, 'BUY');
INSERT INTO Trade VALUES ('T003', 'B001', TO_TIMESTAMP('2022-01-13 10:00:00', 'YYYY-
MM-DD HH24:MI:SS'), 103, 30, 'SELL');
INSERT INTO Trade VALUES ('T004', 'B001', TO_TIMESTAMP('2022-01-14 10:00:00', 'YYYY-
MM-DD HH24:MI:SS'), 104, 40, 'BUY');
INSERT INTO Trade VALUES ('T005', 'B001', TO_TIMESTAMP('2022-01-15 10:00:00', 'YYYY-
MM-DD HH24:MI:SS'), 105, 50, 'SELL');

-- Commit changes
COMMIT;
```