SQL (Generic):

CreateTable_SQL.sql:

```
-- Create table script
CREATE TABLE ExampleTable (
   ID INT PRIMARY KEY,
   FirstName VARCHAR(50),
   LastName VARCHAR(50),
   Age INT
);
```

InsertData_SQL.sql:

```
-- Insert data script
INSERT INTO ExampleTable (ID, FirstName, LastName, Age)
VALUES (1, 'John', 'Doe', 30);

INSERT INTO ExampleTable (ID, FirstName, LastName, Age)
VALUES (2, 'Jane', 'Smith', 25);
```

QueryData_SQL.sql:

```
-- Query data script
SELECT * FROM ExampleTable;
```

UpdateData_SQL.sql:

```
-- Update data script
UPDATE ExampleTable
SET Age = 31
WHERE ID = 1;
```

DeleteData_SQL.sql:

```
-- Delete data script
DELETE FROM ExampleTable
WHERE ID = 2;
```

StoredProcedure_SQL.sql:

```
-- Stored procedure script
CREATE PROCEDURE GetAge(IN id INT, OUT age INT)
BEGIN
   SELECT Age INTO age
   FROM ExampleTable
   WHERE ID = id;
END;
```

Transactions_SQL.sql:

```
-- Transaction script
BEGIN TRANSACTION;

-- SQL statements here

COMMIT;
```

ExceptionHandling_SQL.sql:

```
-- Exception handling script
BEGIN
   -- SQL statements here
```

```
EXCEPTION
    WHEN others THEN
        -- Handle exceptions here
END;
```

DynamicSQL_SQL.sql:

```
-- Dynamic SQL script
DECLARE
    sql_query VARCHAR(100);
    result INT;
BEGIN
    sql_query := 'SELECT COUNT(*) FROM ExampleTable';
    EXECUTE IMMEDIATE sql_query INTO result;
    DBMS_OUTPUT.PUT_LINE('Number of rows: ' || result);
END;
/
```

Oracle:

1. CreateTable_Oracle.sql:

```
-- Create table script for Oracle
CREATE TABLE ExampleTable (
    ID NUMBER PRIMARY KEY,
    FirstName VARCHAR2(50),
    LastName VARCHAR2(50),
    Age NUMBER
);
```

2. InsertData_Oracle.sql:

```
-- Insert data script for Oracle
INSERT INTO ExampleTable (ID, FirstName, LastName, Age)
VALUES (1, 'John', 'Doe', 30);

INSERT INTO ExampleTable (ID, FirstName, LastName, Age)
VALUES (2, 'Jane', 'Smith', 25);
```

3. QueryData_Oracle.sql:

```
-- Query data script for Oracle
SELECT * FROM ExampleTable;
```

4. UpdateData_Oracle.sql:

```
-- Update data script for Oracle
UPDATE ExampleTable
SET Age = 31
WHERE ID = 1;
```

5. DeleteData_Oracle.sql

```
-- Delete data script for Oracle
DELETE FROM ExampleTable
WHERE ID = 2;
```

6. StoredProcedure_Oracle.sql:

```
-- Stored procedure script for Oracle
CREATE OR REPLACE PROCEDURE GetAge(p_id IN NUMBER, p_age OUT NUMBER)
IS
BEGIN
   SELECT Age INTO p_age
   FROM ExampleTable
   WHERE ID = p_id;
END;
/
```

7. Transactions_Oracle.sql:

```
-- Transaction script for Oracle
BEGIN
   -- SQL statements here

   COMMIT;
EXCEPTION
   WHEN others THEN
      ROLLBACK;
END;
```

8. ExceptionHandling_Oracle.sql:

```
-- Exception handling script for Oracle
```

```
BEGIN
   -- SQL statements here
EXCEPTION
   WHEN others THEN
      -- Handle exceptions here
END;
/
```

DynamicSQL_Oracle.sql:

```
-- Dynamic SQL script for Oracle
DECLARE
   sql_query VARCHAR2(100);
   result NUMBER;
BEGIN
   sql_query := 'SELECT COUNT(*) FROM ExampleTable';
   EXECUTE IMMEDIATE sql_query INTO result;
   DBMS_OUTPUT.PUT_LINE('Number of rows: ' || result);
END;
/
```

SQL (Generic):

1. CreateTable_SQL.sql:

This script creates a table named ExampleTable with columns ID, FirstName, LastName, and Age. The ID column is the primary key.

2. InsertData_SQL.sql:

Inserts two rows of data into the ExampleTable table, providing values for the ID, FirstName, LastName, and Age columns.

3. QueryData_SQL.sql:

Retrieves all rows from the ExampleTable table using a simple SELECT * statement.

4. UpdateData_SQL.sql:

Updates the Age column to 31 for the row where ID is 1 in the ExampleTable table.

5. DeleteData_SQL.sql:

Deletes the row where ID is 2 from the ExampleTable table.

6. StoredProcedure_SQL.sql:

Defines a simple stored procedure named GetAge that takes an ID as input and returns the corresponding Age using an OUT parameter.

7. Transactions_SQL.sql:

Begins a transaction, executes SQL statements (which you would replace with your actual statements), and commits the transaction if successful. If an error occurs, it rolls back the transaction.

8. ExceptionHandling_SQL.sql:

Demonstrates basic exception handling. If an exception occurs in the enclosed block, it is caught, and you can handle it accordingly.

9. DynamicSQL_SQL.sql:

Declares a dynamic SQL query, executes it using EXECUTE IMMEDIATE, and captures the result. In this case, it counts the number of rows in the ExampleTable.

Oracle:

1. CreateTable_Oracle.sql:

Creates a table named ExampleTable with columns ID, FirstName, LastName, and Age. The ID column is the primary key.

2. InsertData_Oracle.sql:

Inserts two rows of data into the ExampleTable table, providing values for the ID, FirstName, LastName, and Age columns.

3. QueryData_Oracle.sql:

Retrieves all rows from the ExampleTable table using a simple SELECT * statement.

4. UpdateData_Oracle.sql:

Updates the Age column to 31 for the row where ID is 1 in the ExampleTable table.

5. DeleteData_Oracle.sql:

Deletes the row where ID is 2 from the ExampleTable table.

6. StoredProcedure_Oracle.sql:

Defines a simple stored procedure named GetAge that takes an ID as input and returns the corresponding Age using an OUT parameter.

7. Transactions_Oracle.sql:

Begins a transaction, executes SQL statements (which you would replace with your actual statements), and commits the transaction if successful. If an error occurs, it rolls back the transaction.

8. ExceptionHandling_Oracle.sql:

Demonstrates basic exception handling in Oracle. If an exception occurs in the enclosed block, it is caught, and you can handle it accordingly.

9. DynamicSQL_Oracle.sql:

Declares a dynamic SQL query, executes it using EXECUTE IMMEDIATE, and captures the result. In this case, it counts the number of rows in the ExampleTable.