What is DataBase?!

A database is an organized collection of data, generally stored and accessed electronically from a computer system.
Databases are structured to facilitate the storage, retrieval, modification, and deletion of data in conjunction with various data-processing operations.

Types Of DataBase

1. **Relational Databases (RDBMS)**
2. **Non Relational Databases (N-RDBMS)**

By Eslam Khder

1. Relational Databases (RDBMS)

Structure: Data is organized into tables (relations) with rows and columns.

Examples: MySQL, PostgreSQL, Microsoft SQL Server, and Oracle Database.

Use Cases: Suitable for applications requiring complex queries, transaction management, and data integrity..

2. NON Relational Databases (NoSQL Databases)

Non Structure: Store data as documents, typically in JSON

Examples: MongoDB, CouchDB.

Use Cases: Suitable for applications with flexible schema requirements, high scalability, and performance needs, such as real-time web applications and big data analytics.

We will use Relational Databases (RDBMS) (Oracle DB)

1. Install Oracle DB

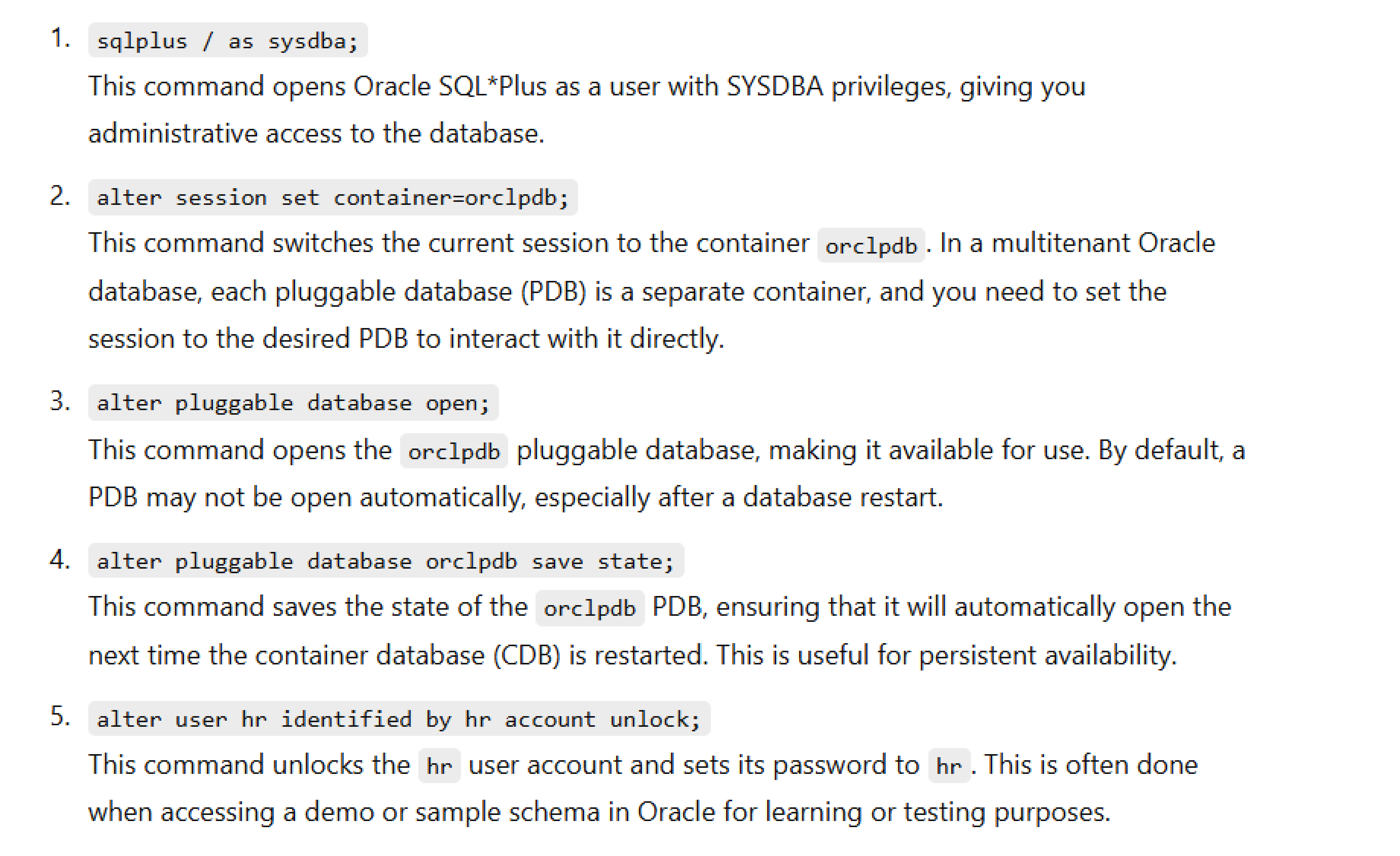
2. Download tool DBeaver

Unlocking the HR Schema

Run this query on commend

```
sqlplus / as sysdba;
alter session set container=orclpdb;
alter pluggable database open;
alter pluggable database orclpdb save state;
alter user hr identified by hr account unlock;
```

Unlocking the HR Schema



1. `sqlplus / as sysdba;`
   This command opens Oracle SQL*Plus as a user with SYSDBA privileges, giving you administrative access to the database.

2. `alter session set container=orclpdb;`
   This command switches the current session to the container `orclpdb`. In a multitenant Oracle database, each pluggable database (PDB) is a separate container, and you need to set the session to the desired PDB to interact with it directly.

3. `alter pluggable database open;`
   This command opens the `orclpdb` pluggable database, making it available for use. By default, a PDB may not be open automatically, especially after a database restart.

4. `alter pluggable database orclpdb save state;`
   This command saves the state of the `orclpdb` PDB, ensuring that it will automatically open the next time the container database (CDB) is restarted. This is useful for persistent availability.

5. `alter user hr identified by hr account unlock;`
   This command unlocks the `hr` user account and sets its password to `hr`. This is often done when accessing a demo or sample schema in Oracle for learning or testing purposes.

By Eslam Khder

SQL Commands | DDL, DML, DCL

1 - Data Definition Language (DDL) Statements

2 - Data Manipulation Language (DML) Statements

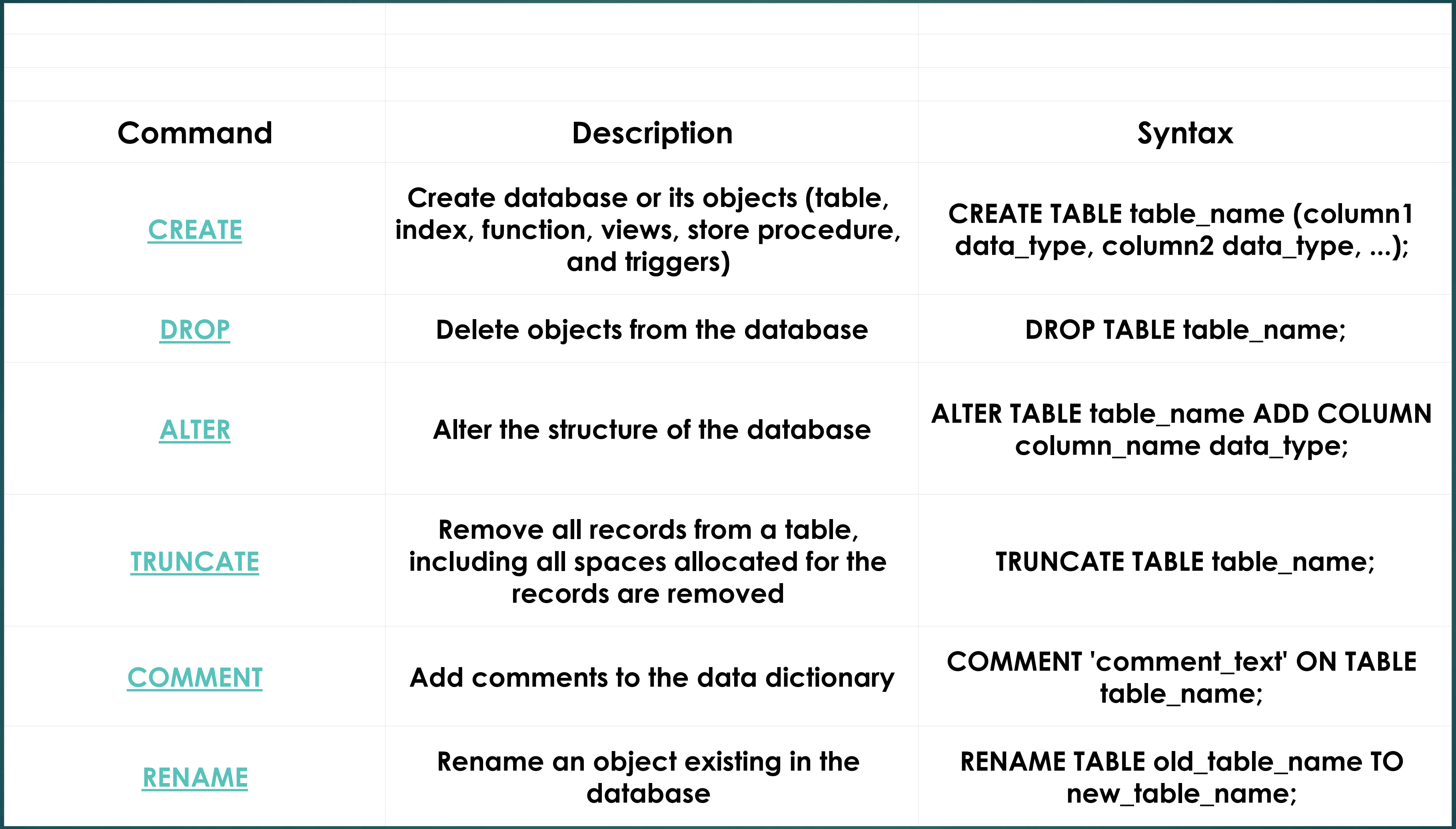
3 - Data Control Language (DCL)

By Eslam Khder

Data Definition Language (DDL) Statements

DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in the database.

Data Definition Language (DDL) Statements



Command	Description	Syntax
CREATE	Create database or its objects (table, index, function, views, store procedure, and triggers)	CREATE TABLE table_name (column1 data_type, column2 data_type, ...);
DROP	Delete objects from the database	DROP TABLE table_name;
ALTER	Alter the structure of the database	ALTER TABLE table_name ADD COLUMN column_name data_type;
TRUNCATE	Remove all records from a table, including all spaces allocated for the records are removed	TRUNCATE TABLE table_name;
COMMENT	Add comments to the data dictionary	COMMENT 'comment_text' ON TABLE table_name;
RENAME	Rename an object existing in the database	RENAME TABLE old_table_name TO new_table_name;

EraaSoft

By Eslam Khder

EraaSoft

SQL Commands | DDL, DML, DCL

1 - Data Definition Language (DDL) Statements

2 - Data Manipulation Language (DML) Statements

3 - Data Control Language (DCL)

By Eslam Khder

Data Manipulation Language (DML) Statements

The SQL commands that deal with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements.

Data Manipulation Language (DML) Statements

Command	Description	Syntax
INSERT	Insert data into a table	INSERT INTO table_name (column1, column2, ...) VALUES (value1, value2, ...);
UPDATE	Update existing data within a table	UPDATE table_name SET column1 = value1, column2 = value2 WHERE condition;
DELETE	Delete records from a database table	DELETE FROM table_name WHERE condition;
LOCK	Table control concurrency	LOCK TABLE table_name IN lock_mode;
CALL	Call a PL/SQL or JAVA subprogram	CALL procedure_name(arguments);
EXPLAIN PLAN	Describe the access path to data	EXPLAIN PLAN FOR SELECT * FROM table_name;

By Eslam Khder

DCL (Data Control Language)

DCL includes commands such as GRANT and REVOKE which mainly deal with the rights, permissions, and other controls of the database system.

Command	Description	Syntax
GRANT	Assigns new privileges to a user account, allowing access to specific database objects, actions, or functions.	GRANT privilege_type [(column_list)] ON [object_type] object_name TO user [WITH GRANT OPTION];
REVOKE	Removes previously granted privileges from a user account, taking away their access to certain database objects or actions.	REVOKE [GRANT OPTION FOR] privilege_type [(column_list)] ON [object_type] object_name FROM user [CASCADE];

By Eslam Khder

Data Definition Language (DDL) Statements

Database Object Naming Rules

Database object names must follow some standard rules

1. They should start with a letter.

2. Can contain only A-Z, a-z, 0-9, -, $, and # characters.

3. Can be up to 128 characters in length.12c Release 2

4. Cannot have the same name as another existing object in the same schema.
5. Cannot be a reserved word like SELECT, FROM, UPDATE, DELETE, WHERE, HAVING, etc.

Data Definition Language (DDL) Statements

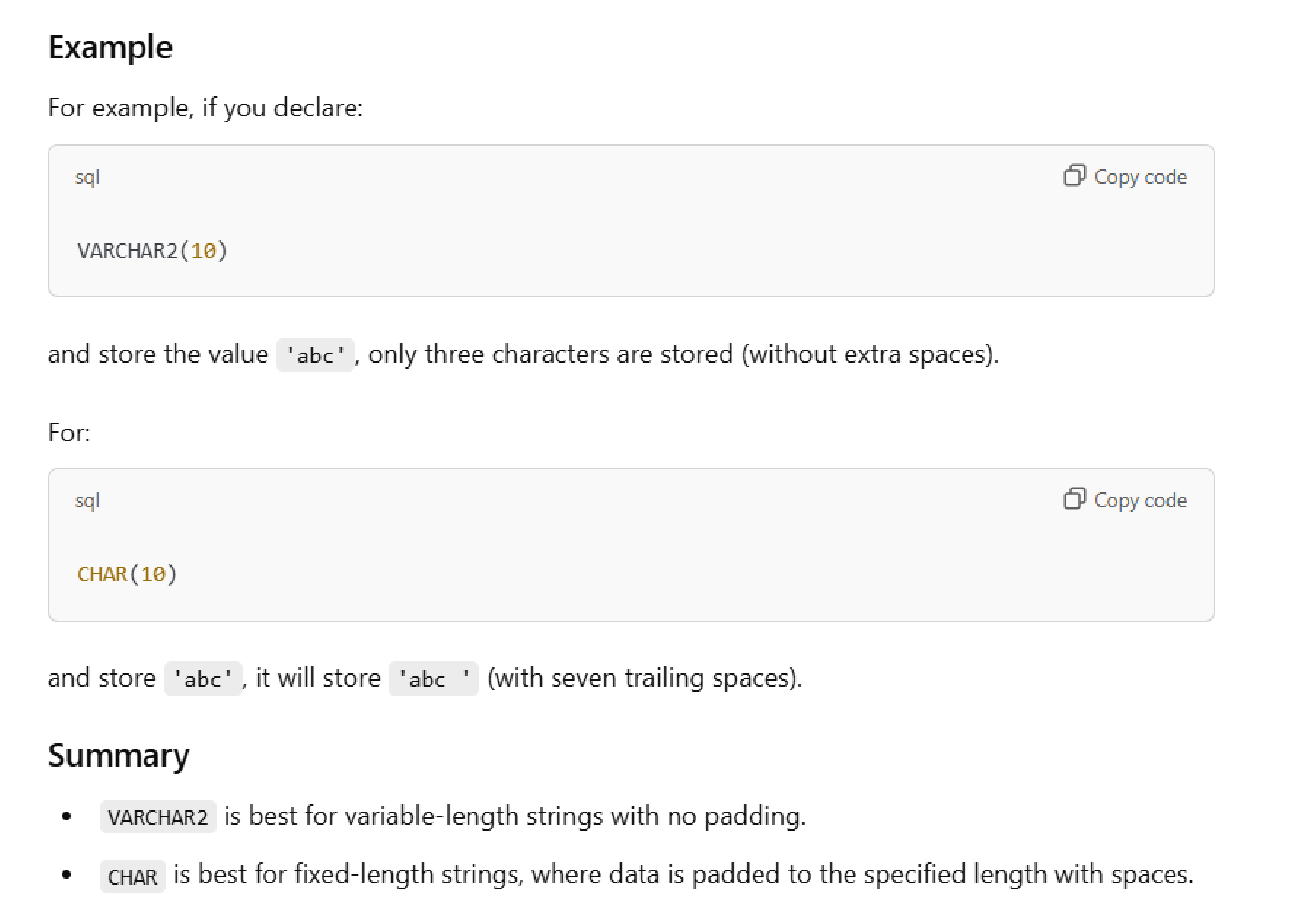
DataTypes

Data types	DESC
VARCHAR2(size)	Variable-length character data
CHAR(size)	Fixed-length character data
NUMBER(p, s)	numeric data (precision, scale)
DATE	Date and time values
CLOB	TO Store FILE

By Eslam Khder

Data Definition Language (DDL) Statements

DataTypes



Example

For example, if you declare:

```sql
VARCHAR2(10)
```

and store the value `'abc'`, only three characters are stored (without extra spaces).

For:

```sql
CHAR(10)
```

and store `'abc'`, it will store `'abc '` (with seven trailing spaces).

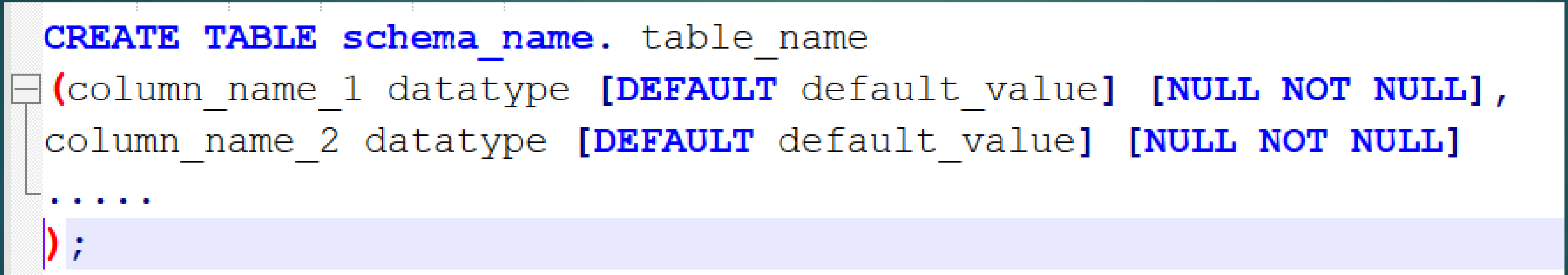
Summary

- `VARCHAR2` is best for variable-length strings with no padding.
- `CHAR` is best for fixed-length strings, where data is padded to the specified length with spaces.

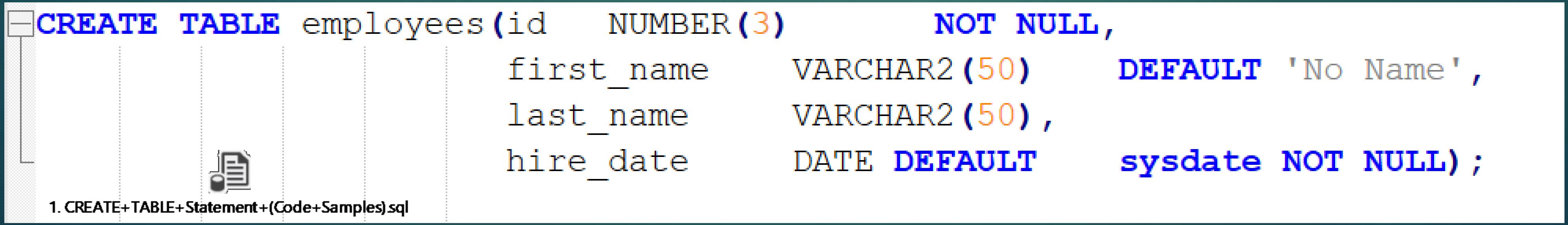
Data Definition Language (DDL) Statements

CREATE TABLE Statement

The CREATE TABLE statement is used to create a new table
To create a table you must have the CREATE TABLE privilege.



```
CREATE TABLE schema_name. table_name
(column_name_1 datatype [DEFAULT default_value] [NULL NOT NULL],
column_name_2 datatype [DEFAULT default_value] [NULL NOT NULL]
.....
);
```

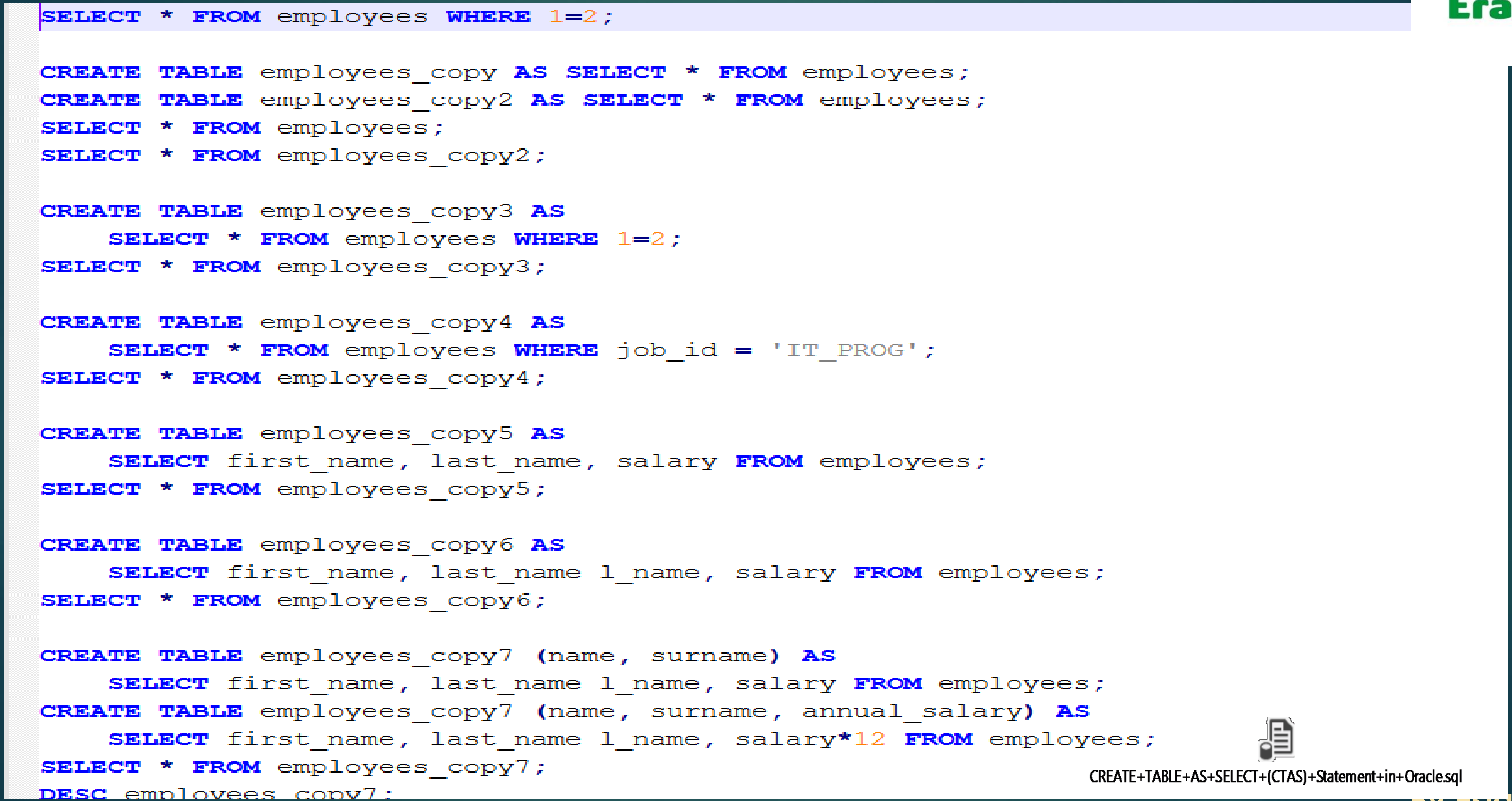


```
CREATE TABLE employees(id   NUMBER(3)       NOT NULL,
                       first_name   VARCHAR2(50)    DEFAULT 'No Name',
                       last_name    VARCHAR2(50),
                       hire_date    DATE DEFAULT    sysdate NOT NULL);
```

1. CREATE+TABLE+Statement+(Code+Samples).sql

By Eslam Khder

Data Definition Language (DDL) Statements



```
SELECT * FROM employees WHERE 1=2;

CREATE TABLE employees_copy AS SELECT * FROM employees;
CREATE TABLE employees_copy2 AS SELECT * FROM employees;
SELECT * FROM employees;
SELECT * FROM employees_copy2;

CREATE TABLE employees_copy3 AS
    SELECT * FROM employees WHERE 1=2;
SELECT * FROM employees_copy3;

CREATE TABLE employees_copy4 AS
    SELECT * FROM employees WHERE job_id = 'IT_PROG';
SELECT * FROM employees_copy4;

CREATE TABLE employees_copy5 AS
    SELECT first_name, last_name, salary FROM employees;
SELECT * FROM employees_copy5;

CREATE TABLE employees_copy6 AS
    SELECT first_name, last_name l_name, salary FROM employees;
SELECT * FROM employees_copy6;

CREATE TABLE employees_copy7 (name, surname) AS
    SELECT first_name, last_name l_name, salary FROM employees;
CREATE TABLE employees_copy7 (name, surname, annual_salary) AS
    SELECT first_name, last_name l_name, salary*12 FROM employees;
SELECT * FROM employees_copy7;
DESC employees_copy7;
```

CREATE+TABLE+AS+SELECT+(CTAS)+Statement+in+Oracle.sql

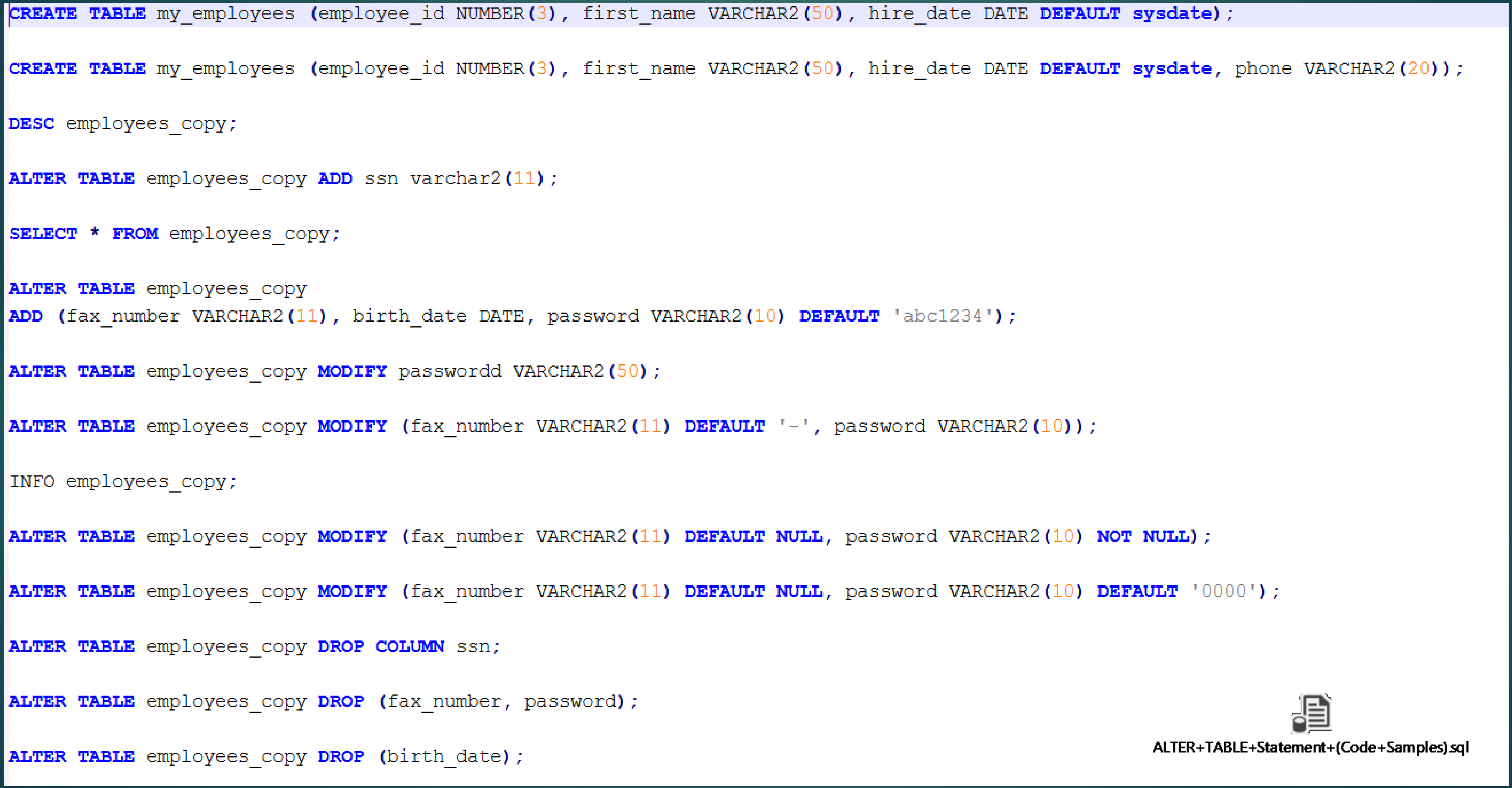
Data Definition Language (DDL) Statements

ALTER TABLE Statements

The ALTER TABLE statement changes the structure of an existing table.
With the ALTER TABLE command, you can:
* Add one or more new columns to a table.
* Modify the data type of one or more existing columns.
* Drop one or more columns from a table.
* Rename a column or a table.
Much more..

By Eslam Khder

Data Definition Language (DDL) Statements



```
CREATE TABLE my_employees (employee_id NUMBER(3), first_name VARCHAR2(50), hire_date DATE DEFAULT sysdate);

CREATE TABLE my_employees (employee_id NUMBER(3), first_name VARCHAR2(50), hire_date DATE DEFAULT sysdate, phone VARCHAR2(20));

DESC employees_copy;

ALTER TABLE employees_copy ADD ssn varchar2(11);

SELECT * FROM employees_copy;

ALTER TABLE employees_copy
ADD (fax_number VARCHAR2(11), birth_date DATE, password VARCHAR2(10) DEFAULT 'abc1234');

ALTER TABLE employees_copy MODIFY passwordd VARCHAR2(50);

ALTER TABLE employees_copy MODIFY (fax_number VARCHAR2(11) DEFAULT '-', password VARCHAR2(10));

INFO employees_copy;

ALTER TABLE employees_copy MODIFY (fax_number VARCHAR2(11) DEFAULT NULL, password VARCHAR2(10) NOT NULL);

ALTER TABLE employees_copy MODIFY (fax_number VARCHAR2(11) DEFAULT NULL, password VARCHAR2(10) DEFAULT '0000');

ALTER TABLE employees_copy DROP COLUMN ssn;

ALTER TABLE employees_copy DROP (fax_number, password);

ALTER TABLE employees_copy DROP (birth_date);
```

ALTER+TABLE+Statement+(Code+Samples).sql

By Eslam Khder

Data Definition Language (DDL) Statements

READ ONLY Tables

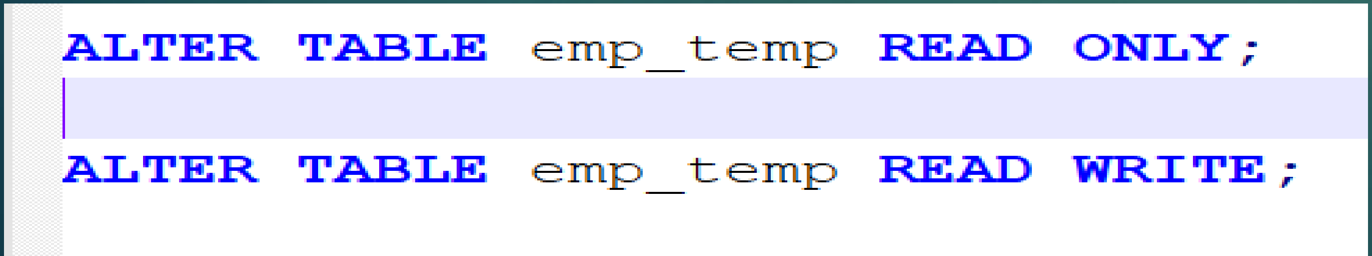
Read-only means allowing users to read, but not modify, data

We need to do maintenance on some tables

During these times, we may want to prevent any DML operations and certain DDL statements that affect the data on those tables against any accidental changes

Oracle allows us to create such tables using the "READ-ONLY" feature.The READ ONLY clause is used at the end of the ALTER TABLE syntax to set a table to read-only

To change a read-only table to read-write again, the READ WRITE clause is used at the end of the ALTER TABLE statement.



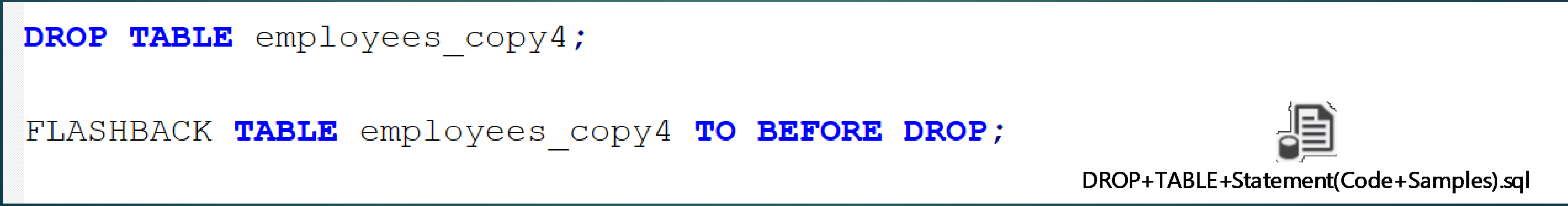
```
ALTER TABLE emp_temp READ ONLY;

ALTER TABLE emp_temp READ WRITE;
```

READ-ONLY+Tables+in+SQL+(Code+Samples).sql

By Eslam Khder

Data Definition Language (DDL) Statements

The DROP TABLE statement removes an existing table with all its data from the database and moves it to the recycle bin
After dropping a table, we can restore it for a short time using the FLASHBACK TABLE statement.
After dropping a table, all the objects related to that table will also be deleted or become invalid.



```sql
DROP TABLE employees_copy4;

FLASHBACK TABLE employees_copy4 TO BEFORE DROP;
```

DROP+TABLE+Statement(Code+Samples).sql

By Eslam Khder

Data Definition Language (DDL) Statements

- TRUNCATE TABLE Statement& The DELETE statement deletes all data row by row whereas the TRUNCATE statement deletes all rowfrom a table more quickly

- The TRUNCATE statement is one of the DDL (DataDefinition Language) statements so it willauto-commit changes immediately after removing data.

- TRUNCATE does not allow rollback.

- The data deleted using the TRUNCATE statement cannot easily be restored (FLASHBACK) becauseTRUNCATE does not generate any undo information or log data.

- The TRUNCATE statement works faster than the DELETE statement.

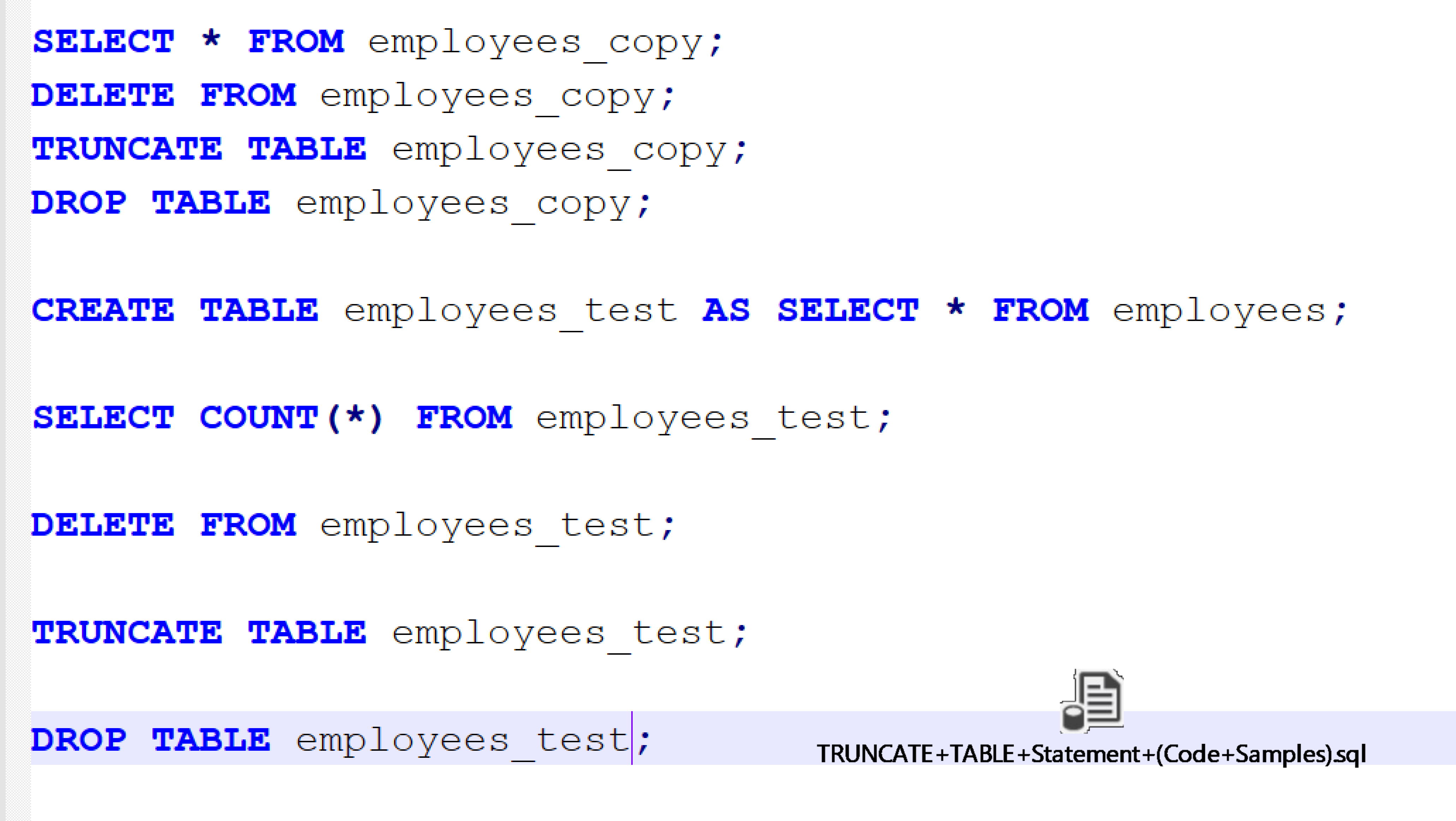
By Eslam Khder

Data Definition Language (DDL) Statements

Why The TRUNCATE statement works faster than the DELETE statement?

By Eslam Khder

Data Definition Language (DDL) Statements



```
SELECT * FROM employees_copy;
DELETE FROM employees_copy;
TRUNCATE TABLE employees_copy;
DROP TABLE employees_copy;

CREATE TABLE employees_test AS SELECT * FROM employees;

SELECT COUNT(*) FROM employees_test;

DELETE FROM employees_test;

TRUNCATE TABLE employees_test;

DROP TABLE employees_test;
```

TRUNCATE+TABLE+Statement+(Code+Samples).sql

By Eslam Khder

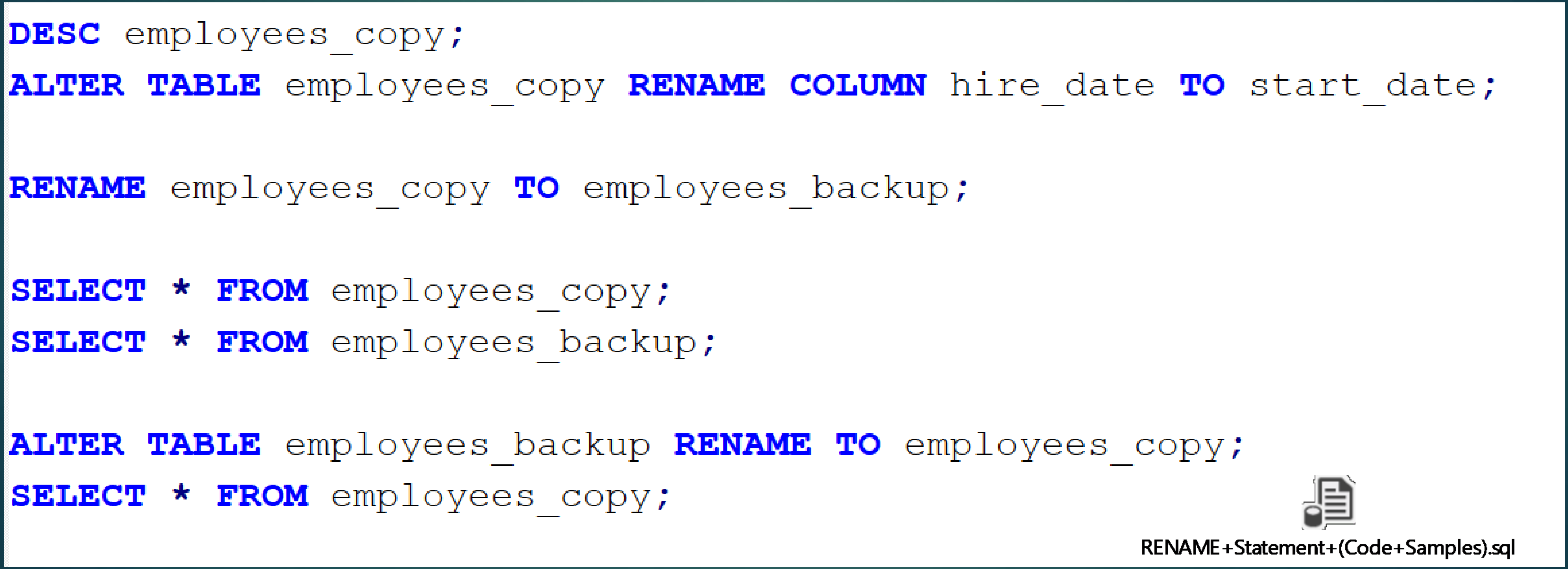
RENAME Statement

The RENAME statement is used to change the name of an existing column or table

We can change the name of a column.

We can change the name of a table.

Data Definition Language (DDL) Statements



```
DESC employees_copy;
ALTER TABLE employees_copy RENAME COLUMN hire_date TO start_date;

RENAME employees_copy TO employees_backup;

SELECT * FROM employees_copy;
SELECT * FROM employees_backup;

ALTER TABLE employees_backup RENAME TO employees_copy;
SELECT * FROM employees_copy;
```

RENAME+Statement+(Code+Samples).sql

Data Definition Language (DML) Statements

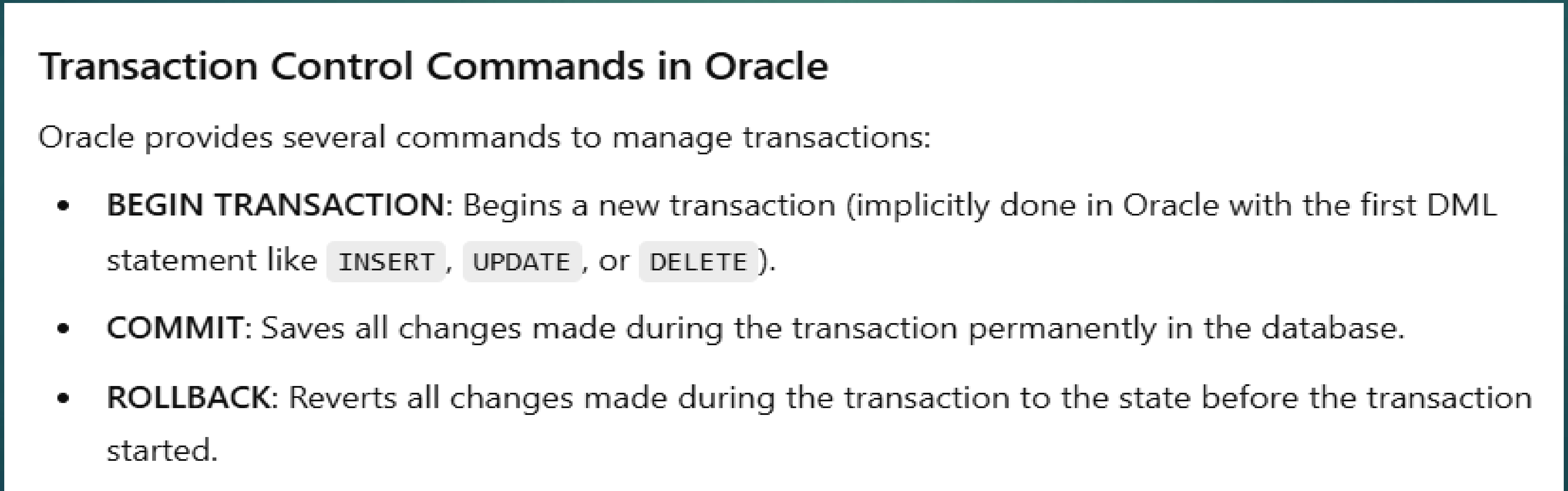
DML is used to add, update, and delete data.

A collection of DML statements is called a transaction.

A transaction starts with the first execution of a DML statement and finishes with a commit or rollback.

Data Definition Language (DML) Statements

transaction is a sequence of one or more SQL operations (such as INSERT, UPDATE, DELETE) executed as a single unit of work.



Transaction Control Commands in Oracle

Oracle provides several commands to manage transactions:

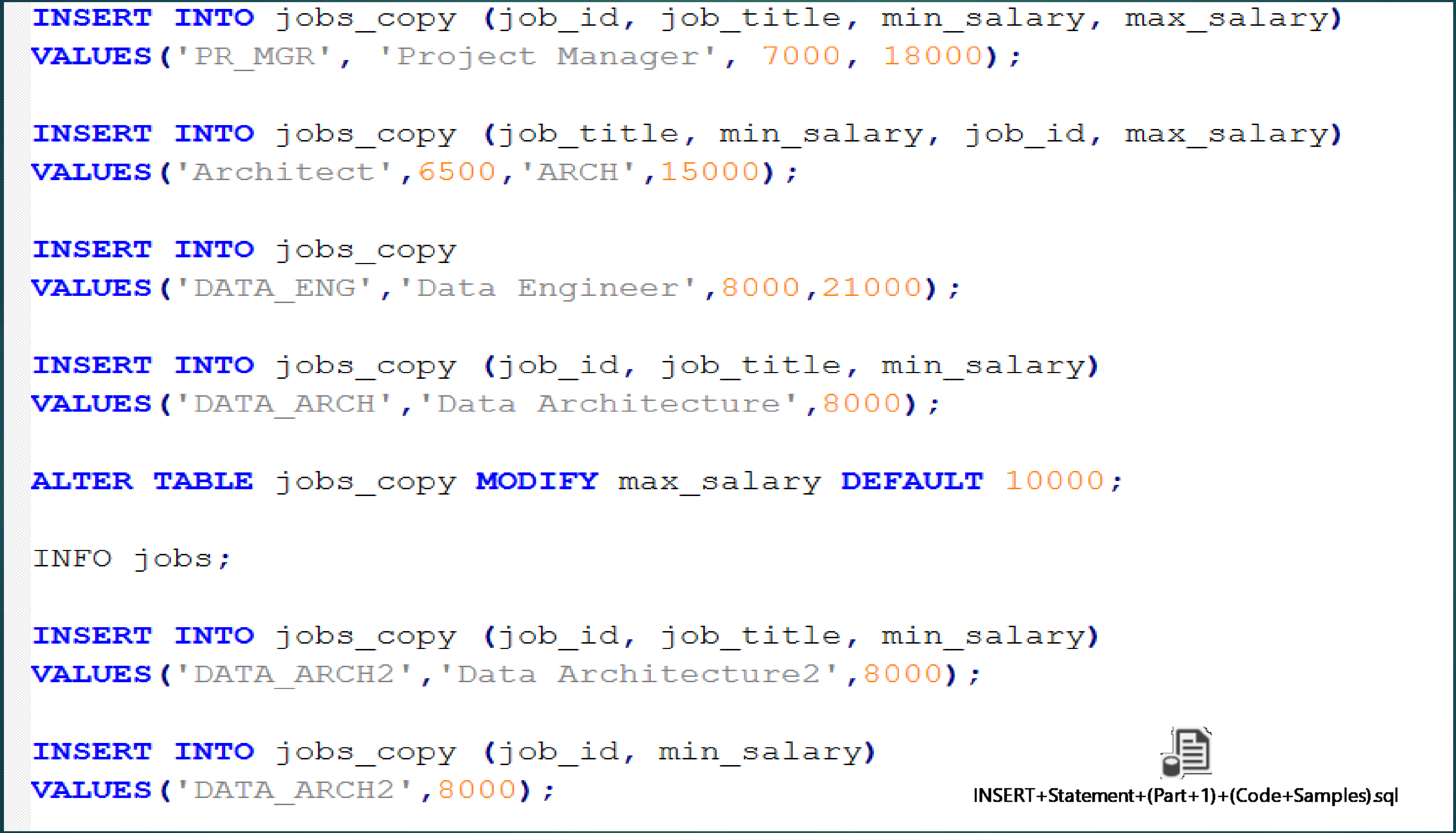
- **BEGIN TRANSACTION**: Begins a new transaction (implicitly done in Oracle with the first DML statement like `INSERT`, `UPDATE`, or `DELETE`).
- **COMMIT**: Saves all changes made during the transaction permanently in the database.
- **ROLLBACK**: Reverts all changes made during the transaction to the state before the transaction started.

INSERT STATEMENT

Use to insert row in table.

EraaSoft

Data Definition Language (DML) Statements

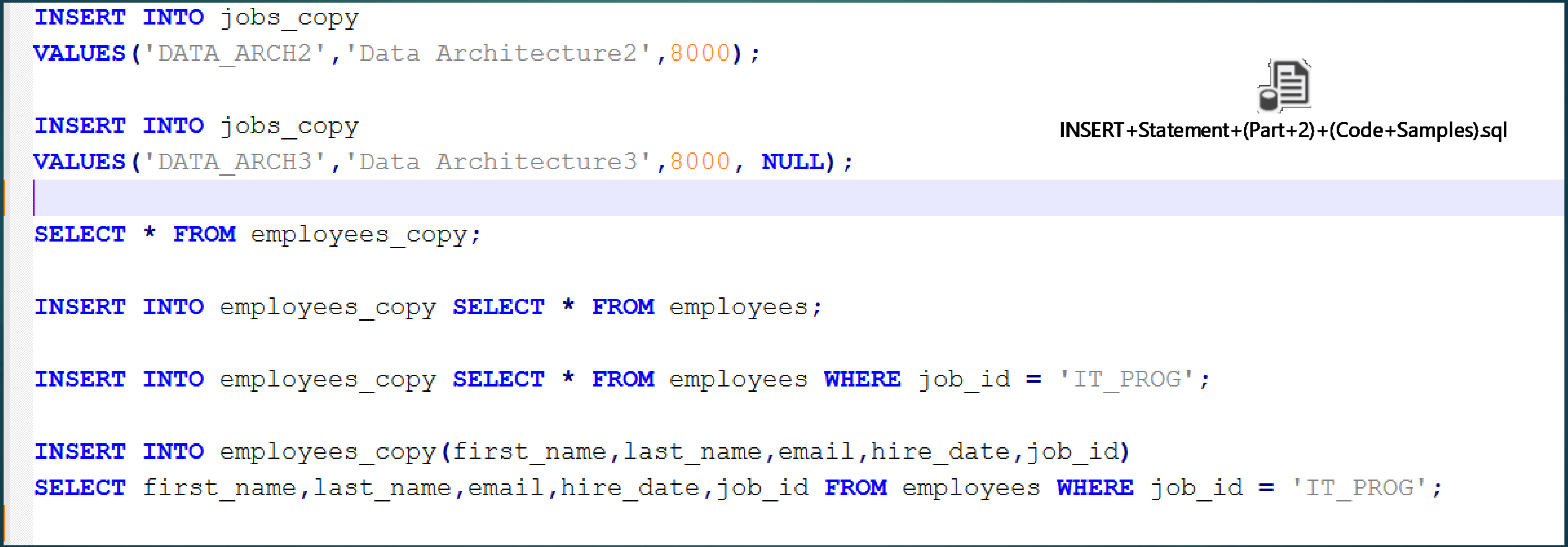


```
INSERT INTO jobs_copy (job_id, job_title, min_salary, max_salary)
VALUES('PR_MGR', 'Project Manager', 7000, 18000);

INSERT INTO jobs_copy (job_title, min_salary, job_id, max_salary)
VALUES('Architect',6500,'ARCH',15000);

INSERT INTO jobs_copy
VALUES('DATA_ENG','Data Engineer',8000,21000);

INSERT INTO jobs_copy (job_id, job_title, min_salary)
VALUES('DATA_ARCH','Data Architecture',8000);

ALTER TABLE jobs_copy MODIFY max_salary DEFAULT 10000;

INFO jobs;

INSERT INTO jobs_copy (job_id, job_title, min_salary)
VALUES('DATA_ARCH2','Data Architecture2',8000);

INSERT INTO jobs_copy (job_id, min_salary)
VALUES('DATA_ARCH2',8000);
```

INSERT+Statement+(Part+1)+(Code+Samples).sql

By Eslam Khder

Data Definition Language (DML) Statements



```
INSERT INTO jobs_copy
VALUES('DATA_ARCH2','Data Architecture2',8000);

INSERT INTO jobs_copy
VALUES('DATA_ARCH3','Data Architecture3',8000, NULL);

SELECT * FROM employees_copy;

INSERT INTO employees_copy SELECT * FROM employees;

INSERT INTO employees_copy SELECT * FROM employees WHERE job_id = 'IT_PROG';

INSERT INTO employees_copy(first_name,last_name,email,hire_date,job_id)
SELECT first_name,last_name,email,hire_date,job_id FROM employees WHERE job_id = 'IT_PROG';
```

INSERT+Statement+(Part+2)+(Code+Samples).sql

By Eslam Khder

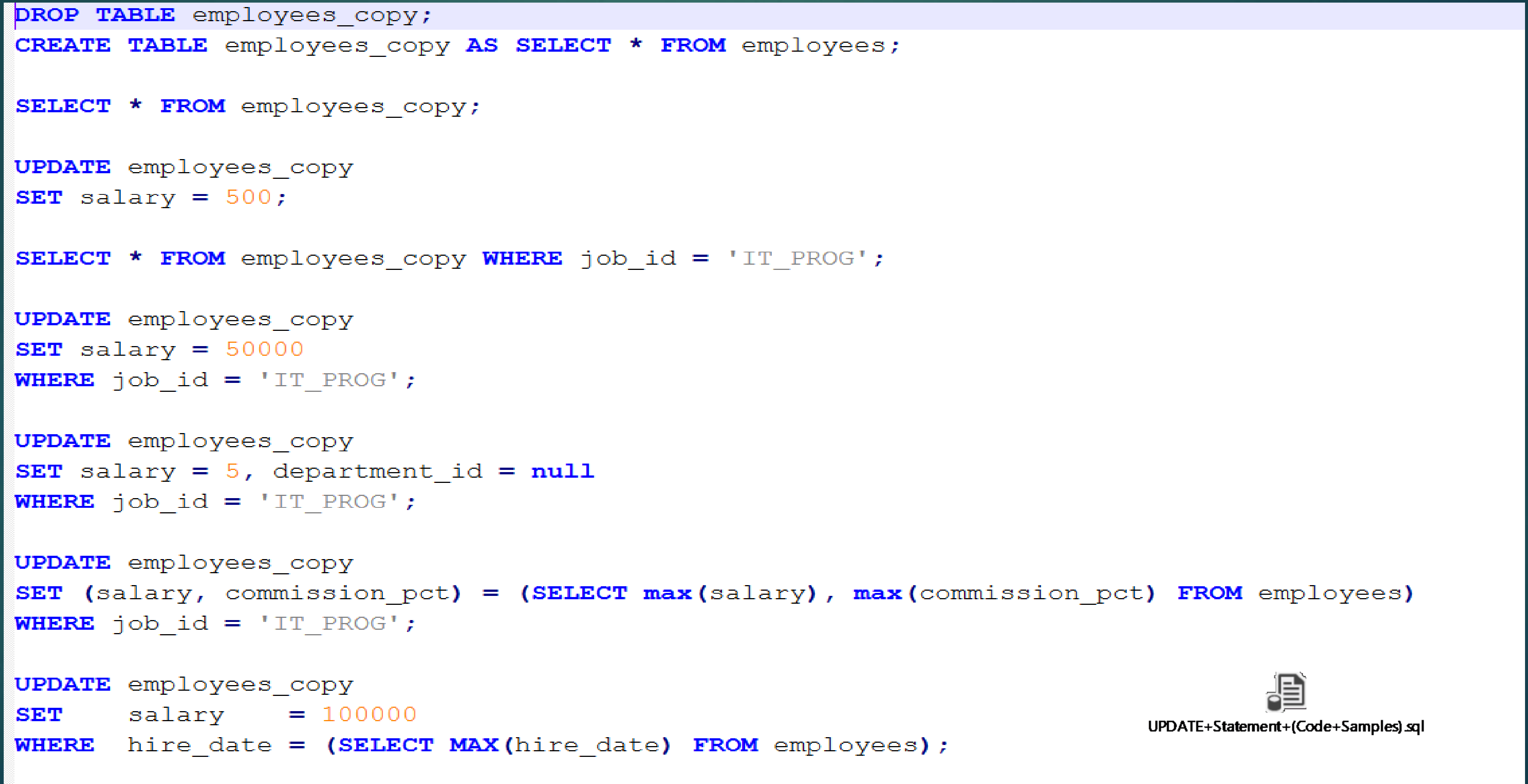
Data Definition Language (DML) Statements

UPDATE STATEMENT

Use to UPDATE row in table.

By Eslam Khder

Data Definition Language (DML) Statements



```sql
DROP TABLE employees_copy;
CREATE TABLE employees_copy AS SELECT * FROM employees;

SELECT * FROM employees_copy;

UPDATE employees_copy
SET salary = 500;

SELECT * FROM employees_copy WHERE job_id = 'IT_PROG';

UPDATE employees_copy
SET salary = 50000
WHERE job_id = 'IT_PROG';

UPDATE employees_copy
SET salary = 5, department_id = null
WHERE job_id = 'IT_PROG';

UPDATE employees_copy
SET (salary, commission_pct) = (SELECT max(salary), max(commission_pct) FROM employees)
WHERE job_id = 'IT_PROG';

UPDATE employees_copy
SET    salary     = 100000
WHERE  hire_date = (SELECT MAX(hire_date) FROM employees);
```

UPDATE+Statement+(Code+Samples).sql

By Eslam Khder

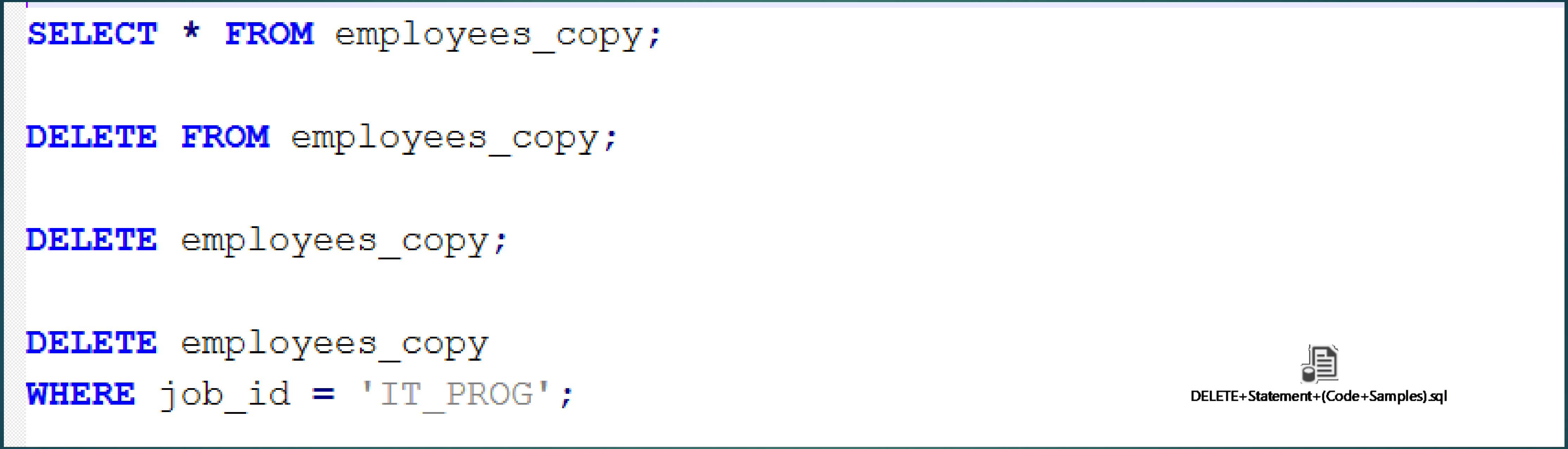
Data Definition Language (DML) Statements

DELETE STATEMENT

Use to DELETE row from table.

By Eslam Khder

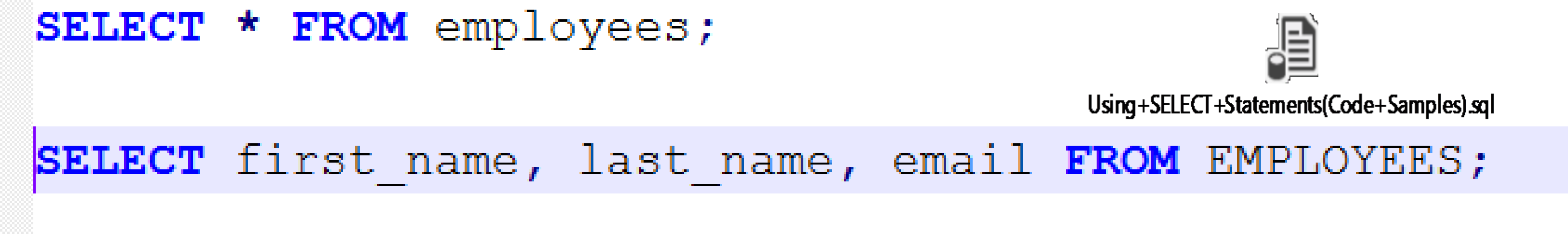
Data Definition Language (DML) Statements



```
SELECT * FROM employees_copy;

DELETE FROM employees_copy;

DELETE employees_copy;

DELETE employees_copy
WHERE job_id = 'IT_PROG';
```

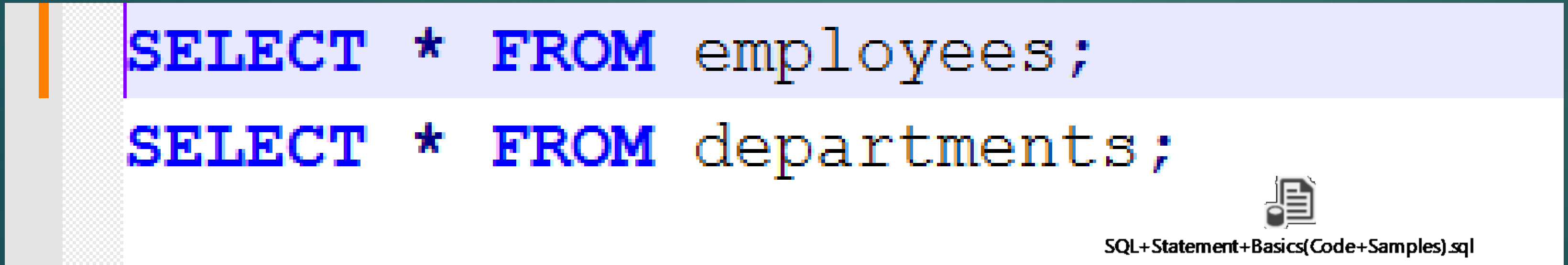
DELETE+Statement+(Code+Samples).sql

Using+SELECT+Statements



```
SELECT * FROM employees;

SELECT first_name, last_name, email FROM EMPLOYEES;
```

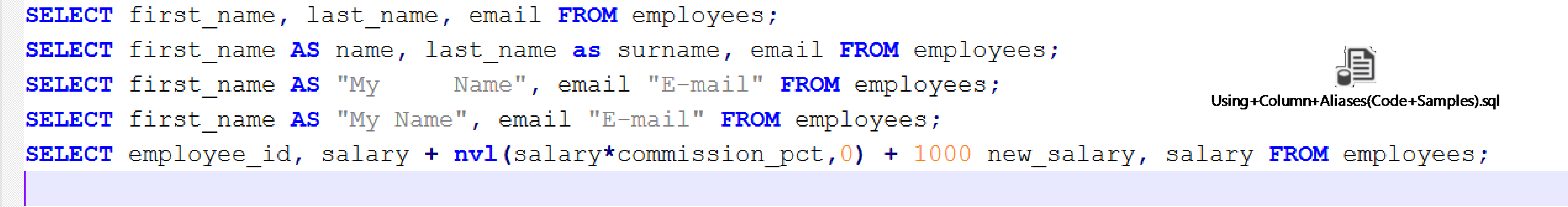
Using+SELECT+Statements(Code+Samples).sql



```
SELECT * FROM employees;
SELECT * FROM departments;
```

SQL+Statement+Basics(Code+Samples).sql

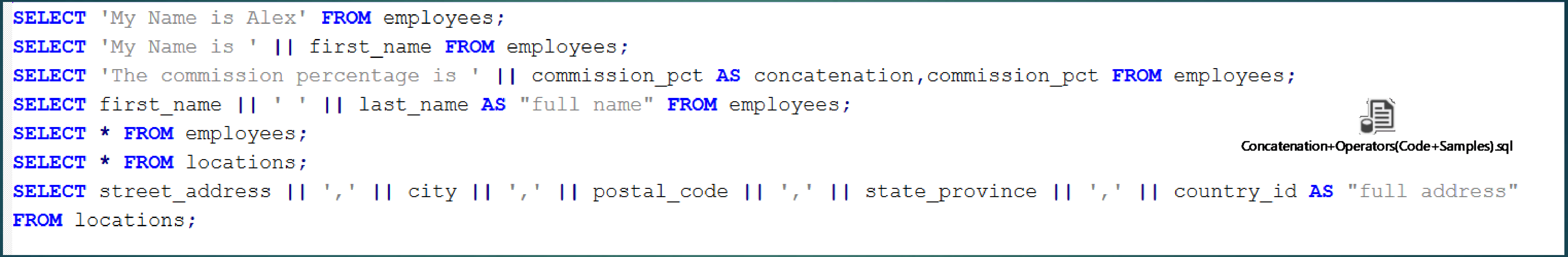
Using Column Aliases



```sql
SELECT first_name, last_name, email FROM employees;
SELECT first_name AS name, last_name as surname, email FROM employees;
SELECT first_name AS "My     Name", email "E-mail" FROM employees;
SELECT first_name AS "My Name", email "E-mail" FROM employees;
SELECT employee_id, salary + nvl(salary*commission_pct,0) + 1000 new_salary, salary FROM employees;
```

Using+Column+Aliases(Code+Samples).sql

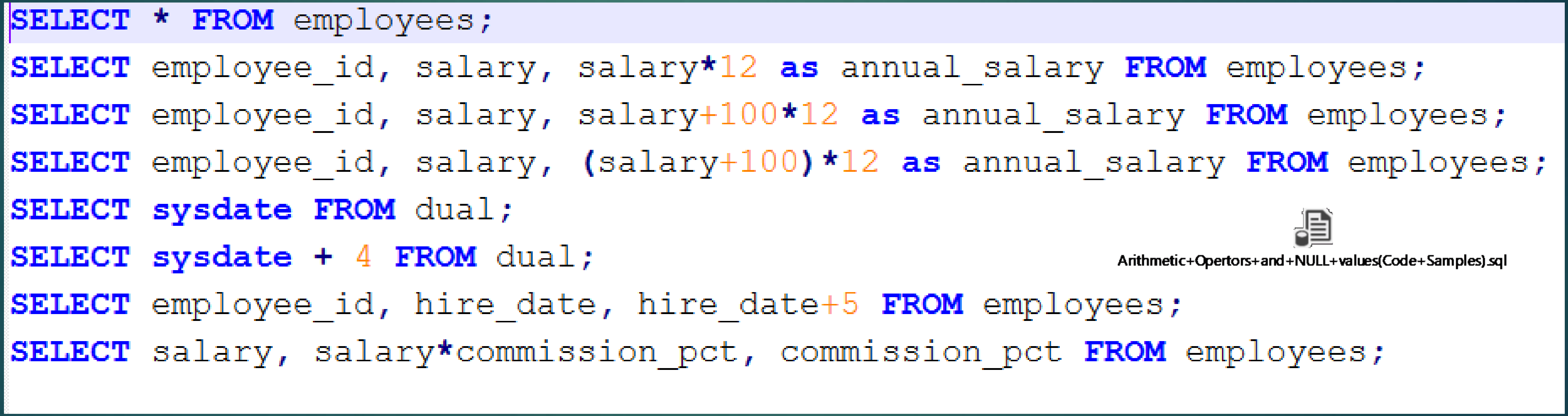
Concatenation Operators



```sql
SELECT 'My Name is Alex' FROM employees;
SELECT 'My Name is ' || first_name FROM employees;
SELECT 'The commission percentage is ' || commission_pct AS concatenation,commission_pct FROM employees;
SELECT first_name || ' ' || last_name AS "full name" FROM employees;
SELECT * FROM employees;
SELECT * FROM locations;
SELECT street_address || ',' || city || ',' || postal_code || ',' || state_province || ',' || country_id AS "full address"
FROM locations;
```

Concatenation+Operators(Code+Samples).sql

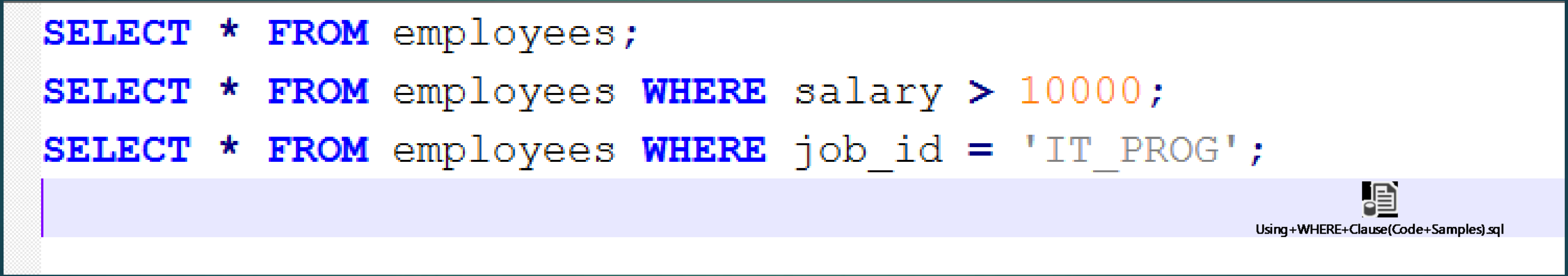
Arithmetic Opertors



```
SELECT * FROM employees;
SELECT employee_id, salary, salary*12 as annual_salary FROM employees;
SELECT employee_id, salary, salary+100*12 as annual_salary FROM employees;
SELECT employee_id, salary, (salary+100)*12 as annual_salary FROM employees;
SELECT sysdate FROM dual;
SELECT sysdate + 4 FROM dual;
SELECT employee_id, hire_date, hire_date+5 FROM employees;
SELECT salary, salary*commission_pct, commission_pct FROM employees;
```

Arithmetic+Opertors+and+NULL+values(Code+Samples).sql

Using WHERE Clause

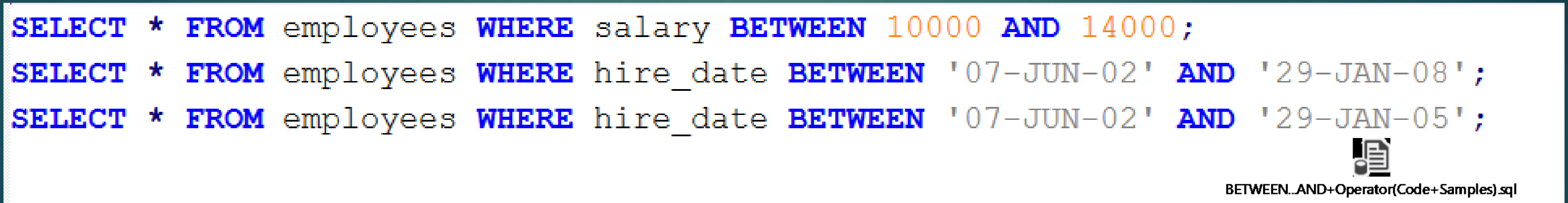


```sql
SELECT * FROM employees;
SELECT * FROM employees WHERE salary > 10000;
SELECT * FROM employees WHERE job_id = 'IT_PROG';
```

Using+WHERE+Clause(Code+Samples).sql

By Eslam Khder

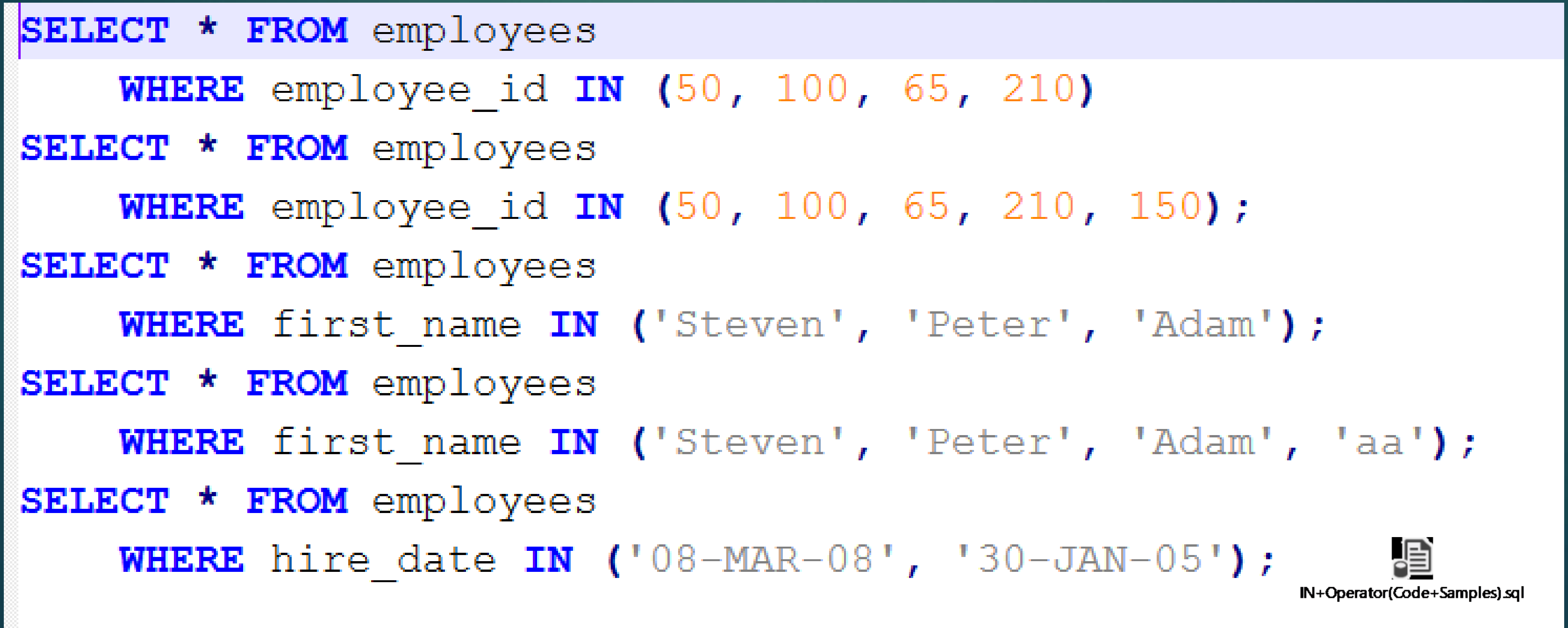
BETWEEN AND Operator



```sql
SELECT * FROM employees WHERE salary BETWEEN 10000 AND 14000;
SELECT * FROM employees WHERE hire_date BETWEEN '07-JUN-02' AND '29-JAN-08';
SELECT * FROM employees WHERE hire_date BETWEEN '07-JUN-02' AND '29-JAN-05';
```

BETWEEN..AND+Operator(Code+Samples).sql

IN Operator



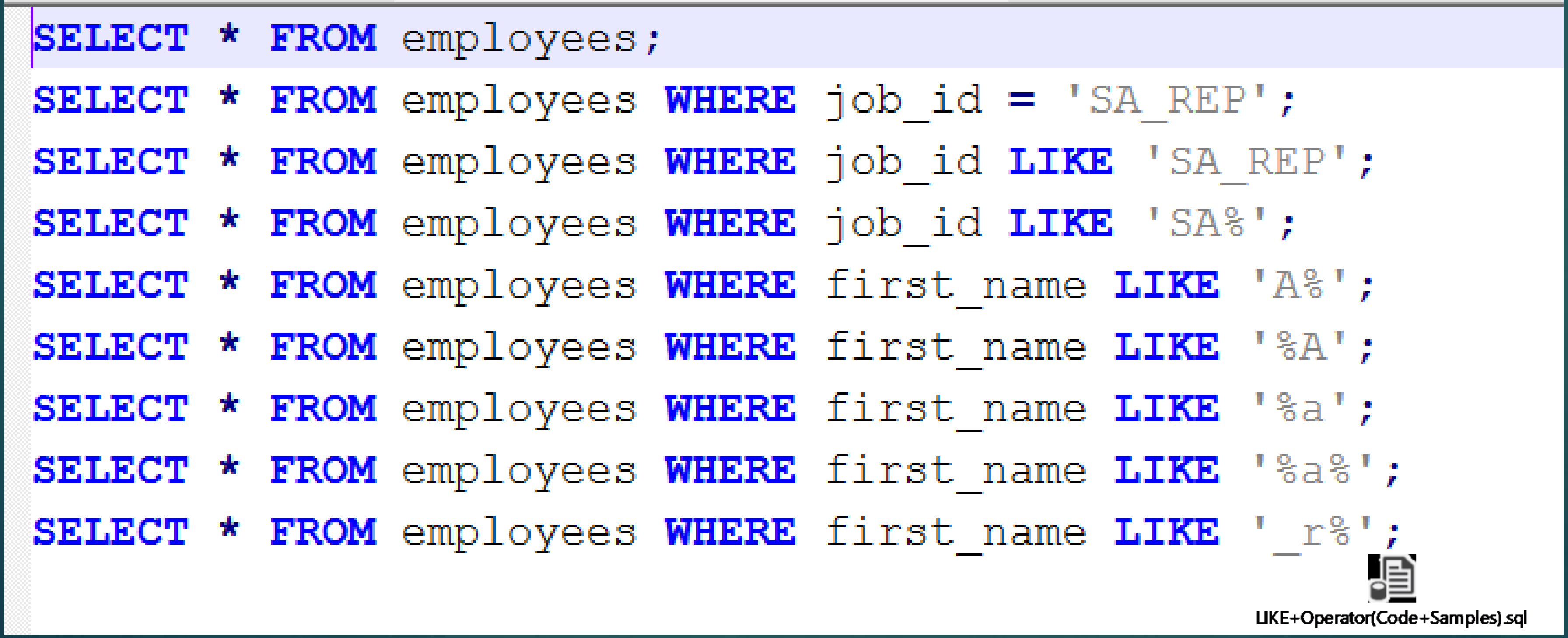
```
SELECT * FROM employees
    WHERE employee_id IN (50, 100, 65, 210)
SELECT * FROM employees
    WHERE employee_id IN (50, 100, 65, 210, 150);
SELECT * FROM employees
    WHERE first_name IN ('Steven', 'Peter', 'Adam');
SELECT * FROM employees
    WHERE first_name IN ('Steven', 'Peter', 'Adam', 'aa');
SELECT * FROM employees
    WHERE hire_date IN ('08-MAR-08', '30-JAN-05');
```

IN+Operator(Code+Samples).sql

By Eslam Khder

EraaSoft

LIKE Operator

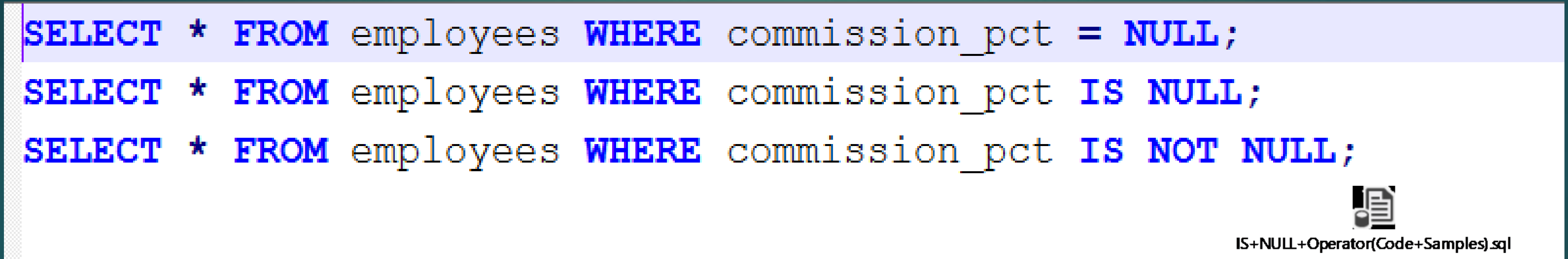


```
SELECT * FROM employees;
SELECT * FROM employees WHERE job_id = 'SA_REP';
SELECT * FROM employees WHERE job_id LIKE 'SA_REP';
SELECT * FROM employees WHERE job_id LIKE 'SA%';
SELECT * FROM employees WHERE first_name LIKE 'A%';
SELECT * FROM employees WHERE first_name LIKE '%A';
SELECT * FROM employees WHERE first_name LIKE '%a';
SELECT * FROM employees WHERE first_name LIKE '%a%';
SELECT * FROM employees WHERE first_name LIKE '_r%';
```

LIKE+Operator(Code+Samples).sql

By Eslam Khder

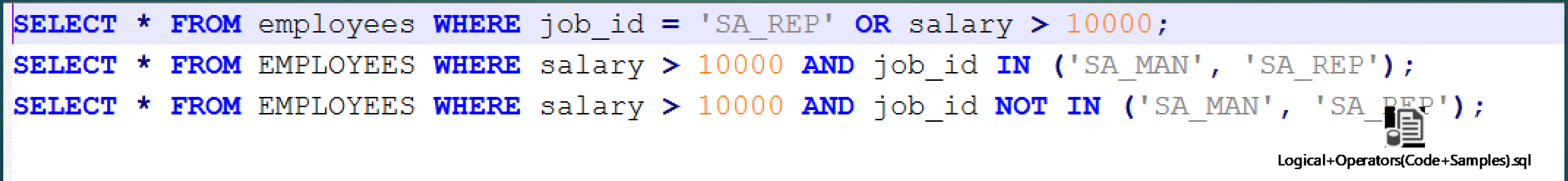
IS NULL Operator



```
SELECT * FROM employees WHERE commission_pct = NULL;
SELECT * FROM employees WHERE commission_pct IS NULL;
SELECT * FROM employees WHERE commission_pct IS NOT NULL;
```

IS+NULL+Operator(Code+Samples).sql

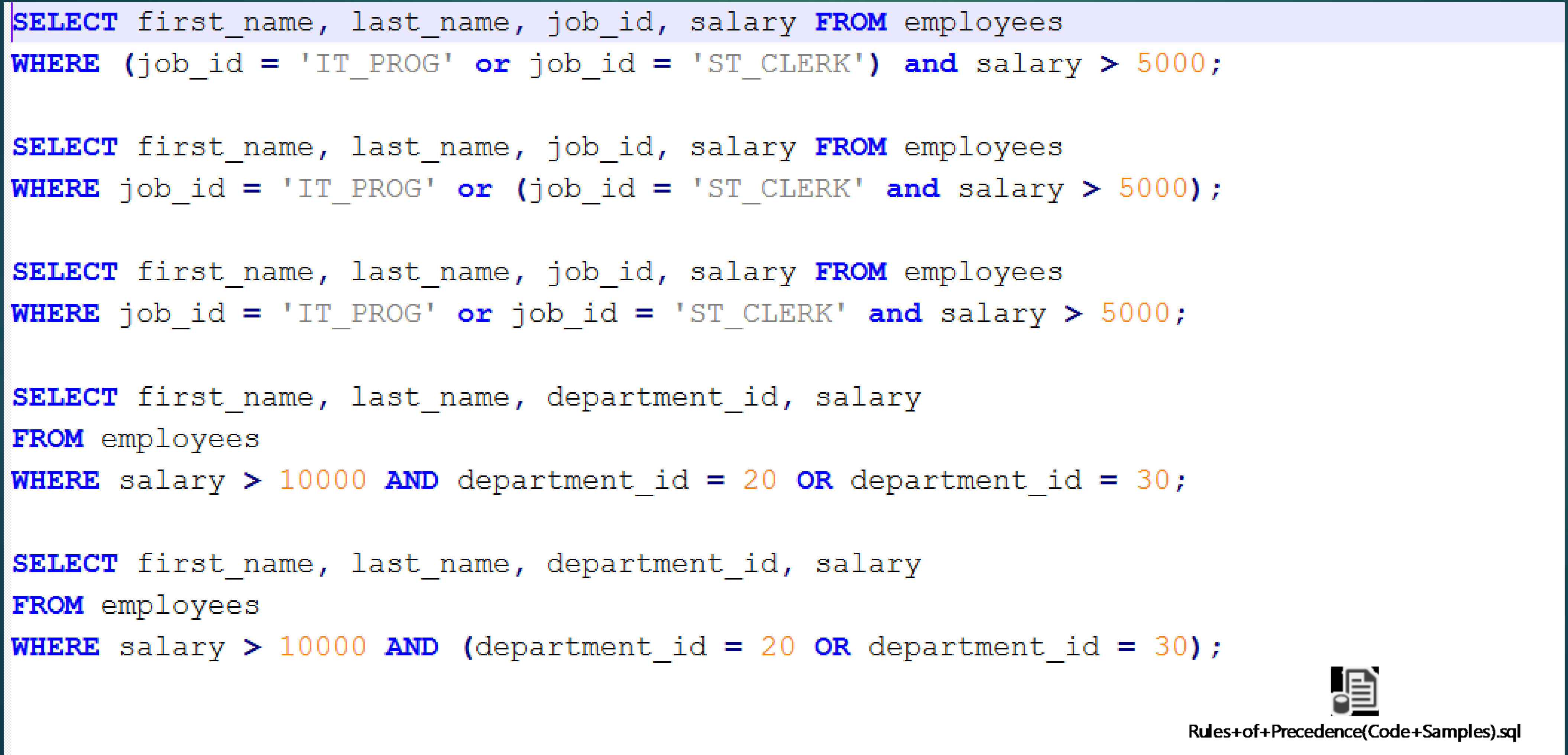
Logical Operators



```
SELECT * FROM employees WHERE job_id = 'SA_REP' OR salary > 10000;
SELECT * FROM EMPLOYEES WHERE salary > 10000 AND job_id IN ('SA_MAN', 'SA_REP');
SELECT * FROM EMPLOYEES WHERE salary > 10000 AND job_id NOT IN ('SA_MAN', 'SA_REP');
```

By Eslam Khder

Rules of Precedence

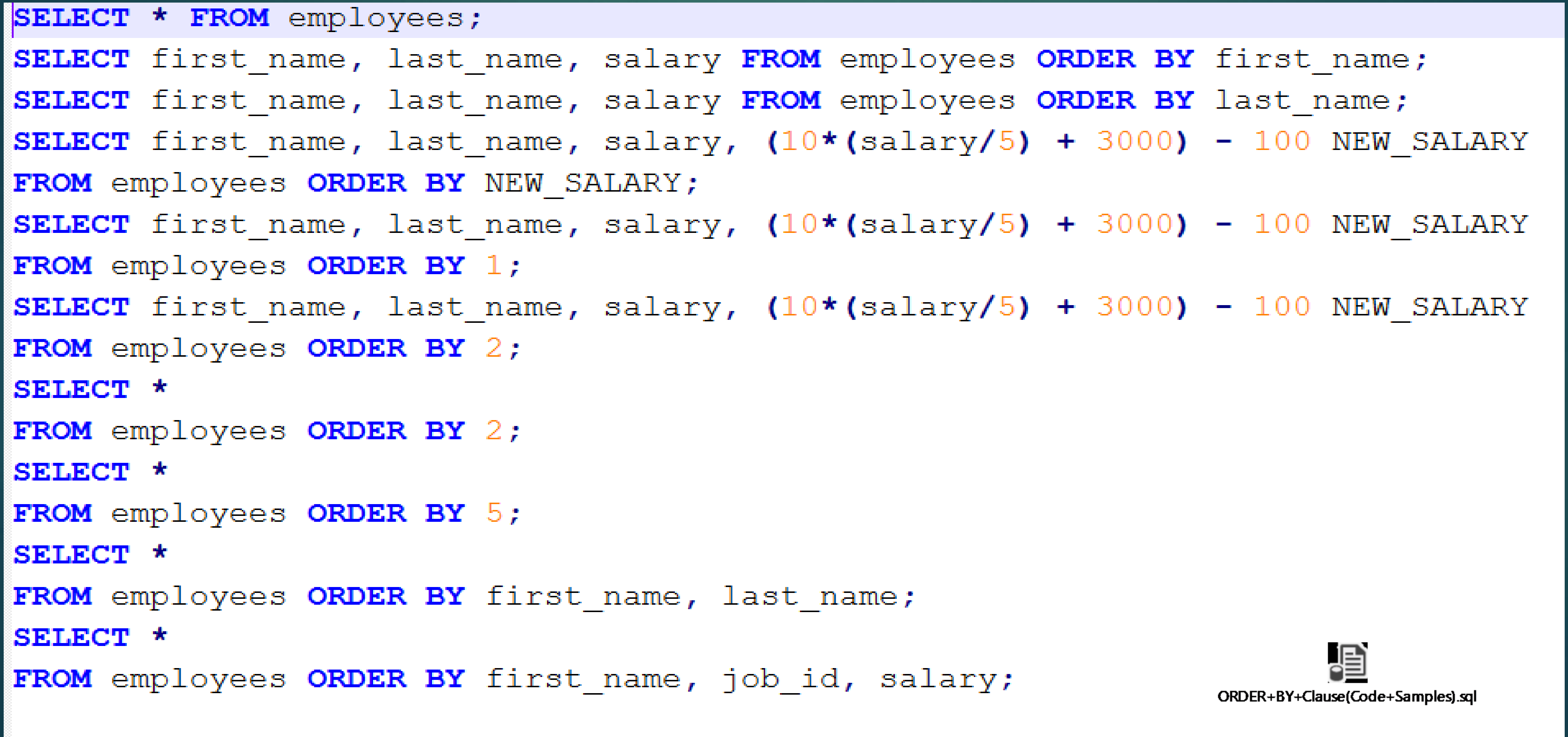


```sql
SELECT first_name, last_name, job_id, salary FROM employees
WHERE (job_id = 'IT_PROG' or job_id = 'ST_CLERK') and salary > 5000;

SELECT first_name, last_name, job_id, salary FROM employees
WHERE job_id = 'IT_PROG' or (job_id = 'ST_CLERK' and salary > 5000);

SELECT first_name, last_name, job_id, salary FROM employees
WHERE job_id = 'IT_PROG' or job_id = 'ST_CLERK' and salary > 5000;

SELECT first_name, last_name, department_id, salary
FROM employees
WHERE salary > 10000 AND department_id = 20 OR department_id = 30;

SELECT first_name, last_name, department_id, salary
FROM employees
WHERE salary > 10000 AND (department_id = 20 OR department_id = 30);
```

Rules+of+Precedence(Code+Samples).sql

By Eslam Khder

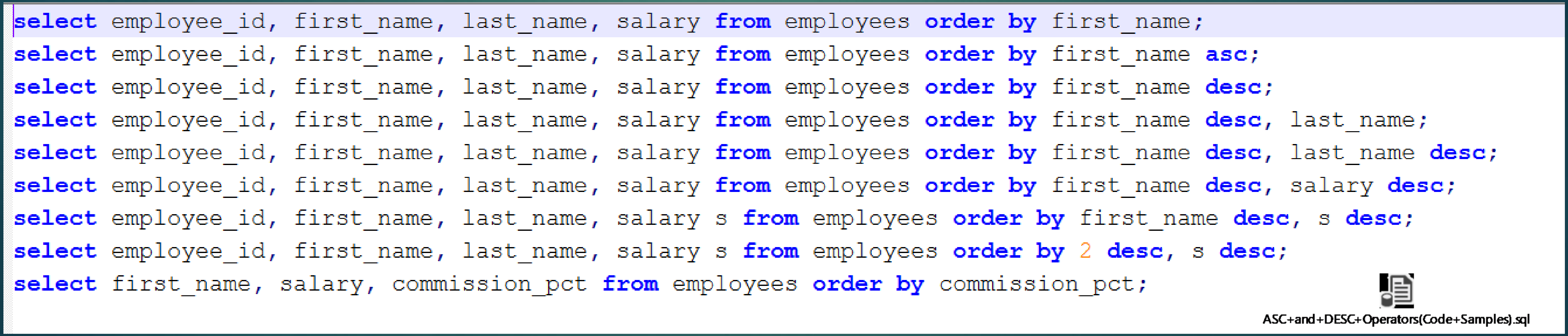
ORDER BY Clause



```
SELECT * FROM employees;
SELECT first_name, last_name, salary FROM employees ORDER BY first_name;
SELECT first_name, last_name, salary FROM employees ORDER BY last_name;
SELECT first_name, last_name, salary, (10*(salary/5) + 3000) - 100 NEW_SALARY
FROM employees ORDER BY NEW_SALARY;
SELECT first_name, last_name, salary, (10*(salary/5) + 3000) - 100 NEW_SALARY
FROM employees ORDER BY 1;
SELECT first_name, last_name, salary, (10*(salary/5) + 3000) - 100 NEW_SALARY
FROM employees ORDER BY 2;
SELECT *
FROM employees ORDER BY 2;
SELECT *
FROM employees ORDER BY 5;
SELECT *
FROM employees ORDER BY first_name, last_name;
SELECT *
FROM employees ORDER BY first_name, job_id, salary;
```

ORDER+BY+Clause(Code+Samples).sql

ASC and DESC Operators

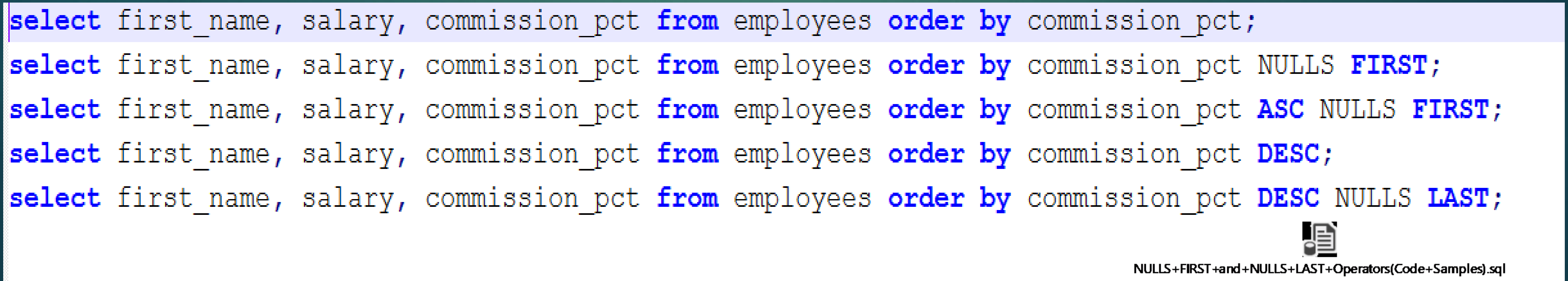


```
select employee_id, first_name, last_name, salary from employees order by first_name;
select employee_id, first_name, last_name, salary from employees order by first_name asc;
select employee_id, first_name, last_name, salary from employees order by first_name desc;
select employee_id, first_name, last_name, salary from employees order by first_name desc, last_name;
select employee_id, first_name, last_name, salary from employees order by first_name desc, last_name desc;
select employee_id, first_name, last_name, salary from employees order by first_name desc, salary desc;
select employee_id, first_name, last_name, salary s from employees order by first_name desc, s desc;
select employee_id, first_name, last_name, salary s from employees order by 2 desc, s desc;
select first_name, salary, commission_pct from employees order by commission_pct;
```

ASC+and+DESC+Operators(Code+Samples).sql

By Eslam Khder

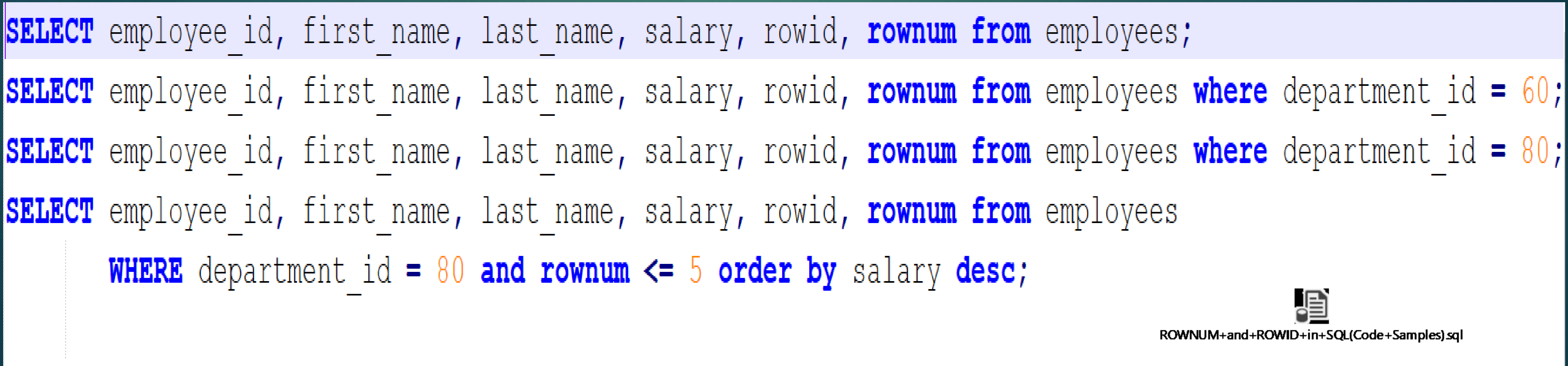
NULLS FIRST and NULLS LAST Operators



```
select first_name, salary, commission_pct from employees order by commission_pct;
select first_name, salary, commission_pct from employees order by commission_pct NULLS FIRST;
select first_name, salary, commission_pct from employees order by commission_pct ASC NULLS FIRST;
select first_name, salary, commission_pct from employees order by commission_pct DESC;
select first_name, salary, commission_pct from employees order by commission_pct DESC NULLS LAST;
```

NULLS+FIRST+and+NULLS+LAST+Operators(Code+Samples).sql

ROWNUM and ROWID



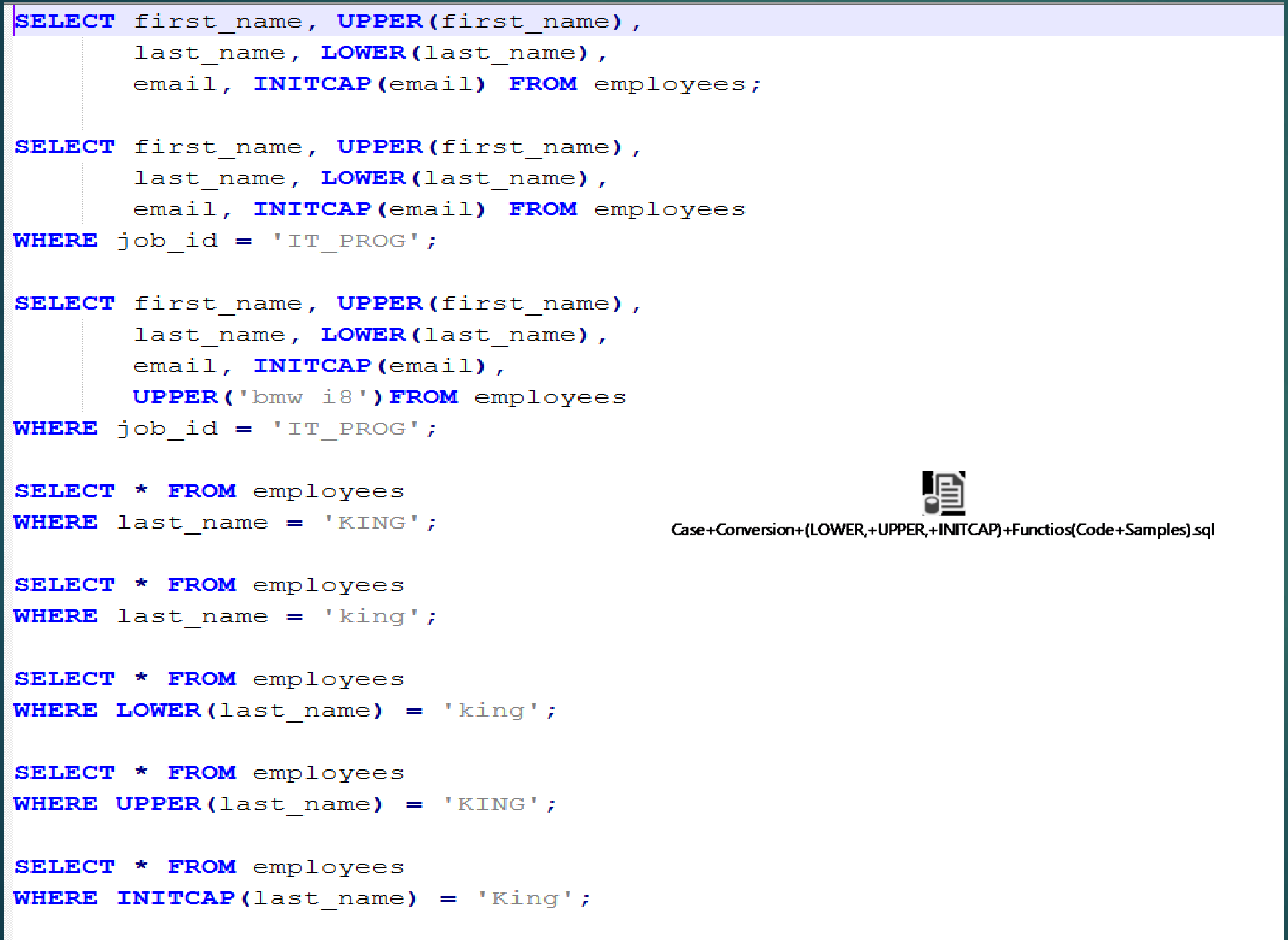
```sql
SELECT employee_id, first_name, last_name, salary, rowid, rownum from employees;
SELECT employee_id, first_name, last_name, salary, rowid, rownum from employees where department_id = 60;
SELECT employee_id, first_name, last_name, salary, rowid, rownum from employees where department_id = 80;
SELECT employee_id, first_name, last_name, salary, rowid, rownum from employees
      WHERE department_id = 80 and rownum <= 5 order by salary desc;
```

ROWNUM+and+ROWID+in+SQL(Code+Samples).sql

By Eslam Khder

Case Conversion (LOWER,+UPPER,+INITCAP)

EraaSoft

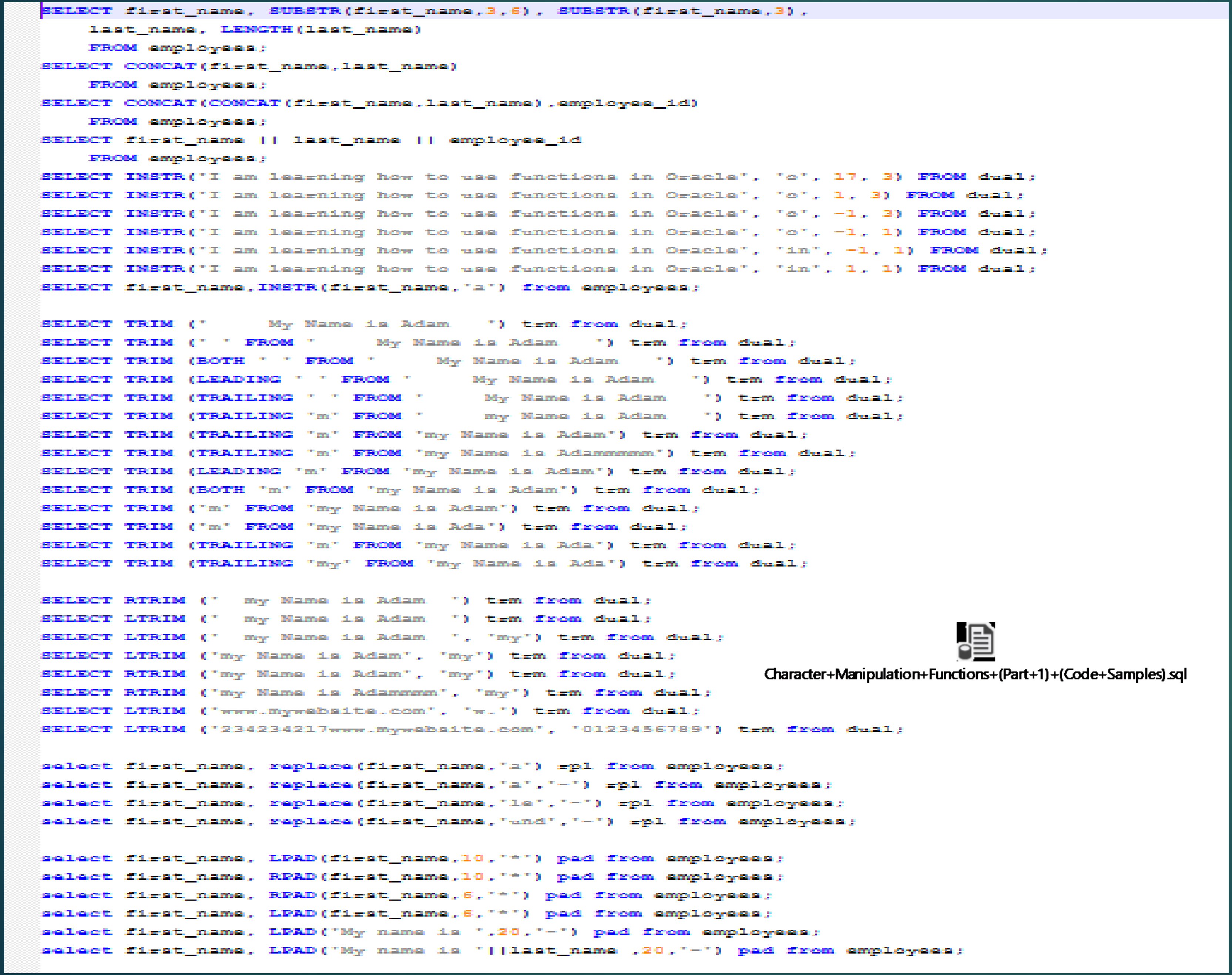


```sql
SELECT first_name, UPPER(first_name),
       last_name, LOWER(last_name),
       email, INITCAP(email) FROM employees;

SELECT first_name, UPPER(first_name),
       last_name, LOWER(last_name),
       email, INITCAP(email) FROM employees
WHERE job_id = 'IT_PROG';

SELECT first_name, UPPER(first_name),
       last_name, LOWER(last_name),
       email, INITCAP(email),
       UPPER('bmw i8')FROM employees
WHERE job_id = 'IT_PROG';

SELECT * FROM employees
WHERE last_name = 'KING';

SELECT * FROM employees
WHERE last_name = 'king';

SELECT * FROM employees
WHERE LOWER(last_name) = 'king';

SELECT * FROM employees
WHERE UPPER(last_name) = 'KING';

SELECT * FROM employees
WHERE INITCAP(last_name) = 'King';
```

Case+Conversion+(LOWER,+UPPER,+INITCAP)+Functios(Code+Samples).sql

By Eslam Khder

Character Manipulation Functions

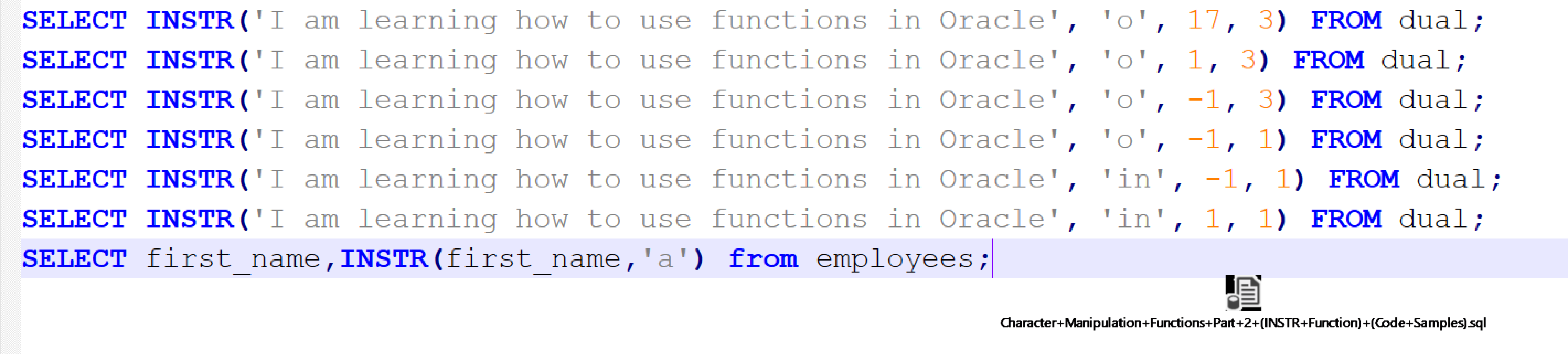


```
SELECT first_name, SUBSTR(first_name,3,6), SUBSTR(first_name,3),
     last_name, LENGTH(last_name)
     FROM employees;
SELECT CONCAT(first_name,last_name)
     FROM employees;
SELECT CONCAT(CONCAT(first_name,last_name),employee_id)
     FROM employees;
SELECT first_name || last_name || employee_id
     FROM employees;
SELECT INSTR('I am learning how to use functions in Oracle', 'o', 17, 3) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'o', 1, 3) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'o', -1, 3) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'o', -1, 1) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'in', -1, 1) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'in', 1, 1) FROM dual;
SELECT first_name,INSTR(first_name,'a') from employees;

SELECT TRIM ('      My Name is Adam    ') trm from dual;
SELECT TRIM (' ' FROM '      My Name is Adam    ') trm from dual;
SELECT TRIM (BOTH ' ' FROM '      My Name is Adam    ') trm from dual;
SELECT TRIM (LEADING ' ' FROM '      My Name is Adam    ') trm from dual;
SELECT TRIM (TRAILING ' ' FROM '      My Name is Adam    ') trm from dual;
SELECT TRIM (TRAILING 'm' FROM '      my Name is Adam    ') trm from dual;
SELECT TRIM (TRAILING 'm' FROM 'my Name is Adam') trm from dual;
SELECT TRIM (TRAILING 'm' FROM 'my Name is Adammmmm') trm from dual;
SELECT TRIM (LEADING 'm' FROM 'my Name is Adam') trm from dual;
SELECT TRIM (BOTH 'm' FROM 'my Name is Adam') trm from dual;
SELECT TRIM ('m' FROM 'my Name is Adam') trm from dual;
SELECT TRIM ('m' FROM 'my Name is Ada') trm from dual;
SELECT TRIM (TRAILING 'm' FROM 'my Name is Ada') trm from dual;
SELECT TRIM (TRAILING 'my' FROM 'my Name is Ada') trm from dual;

SELECT RTRIM ('  my Name is Adam   ') trm from dual;
SELECT LTRIM ('  my Name is Adam   ') trm from dual;
SELECT LTRIM ('  my Name is Adam  ', 'my') trm from dual;
SELECT LTRIM ('my Name is Adam', 'my') trm from dual;
SELECT RTRIM ('my Name is Adam', 'my') trm from dual;
SELECT RTRIM ('my Name is Adammmm', 'my') trm from dual;
SELECT LTRIM ('www.mywebsite.com', 'w.') trm from dual;
SELECT LTRIM ('234234217www.mywebsite.com', '0123456789') trm from dual;

select first_name, replace(first_name,'a') rpl from employees;
select first_name, replace(first_name,'a','-') rpl from employees;
select first_name, replace(first_name,'le','-') rpl from employees;
select first_name, replace(first_name,'und','-') rpl from employees;

select first_name, LPAD(first_name,10,'*') pad from employees;
select first_name, RPAD(first_name,10,'*') pad from employees;
select first_name, RPAD(first_name,6,'*') pad from employees;
select first_name, LPAD(first_name,6,'*') pad from employees;
select first_name, LPAD('My name is ',20,'-') pad from employees;
select first_name, LPAD('My name is '||last_name ,20,'-') pad from employees;
```

Character+Manipulation+Functions+(Part+1)+(Code+Samples).sql

By Eslam Khder

INSTR Function

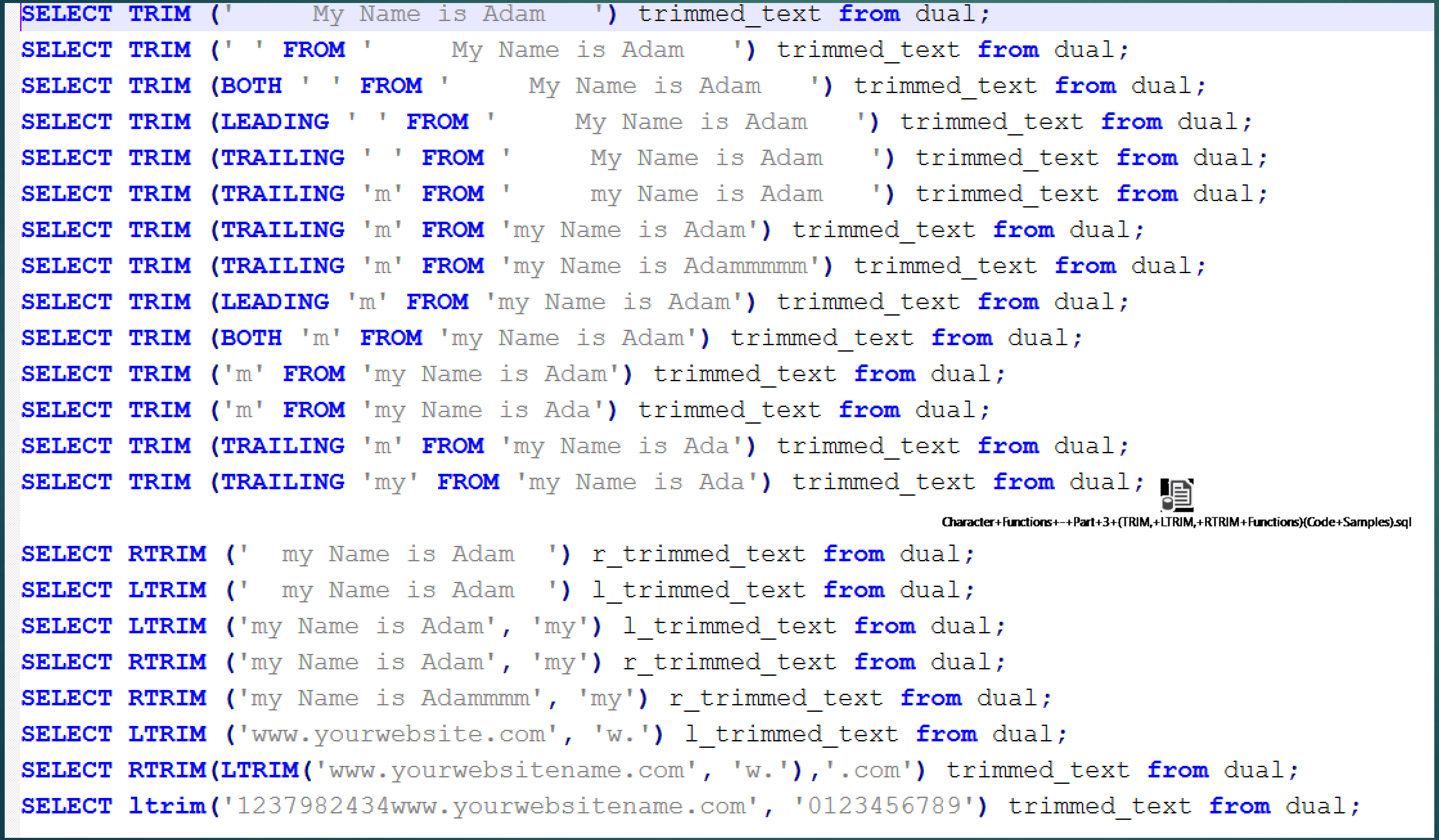


```
SELECT INSTR('I am learning how to use functions in Oracle', 'o', 17, 3) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'o', 1, 3) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'o', -1, 3) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'o', -1, 1) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'in', -1, 1) FROM dual;
SELECT INSTR('I am learning how to use functions in Oracle', 'in', 1, 1) FROM dual;
SELECT first_name,INSTR(first_name,'a') from employees;
```

Character+Manipulation+Functions+Part+2+(INSTR+Function)+(Code+Samples).sql

By Eslam Khder

TRIM LTRIM RTRIM Functions

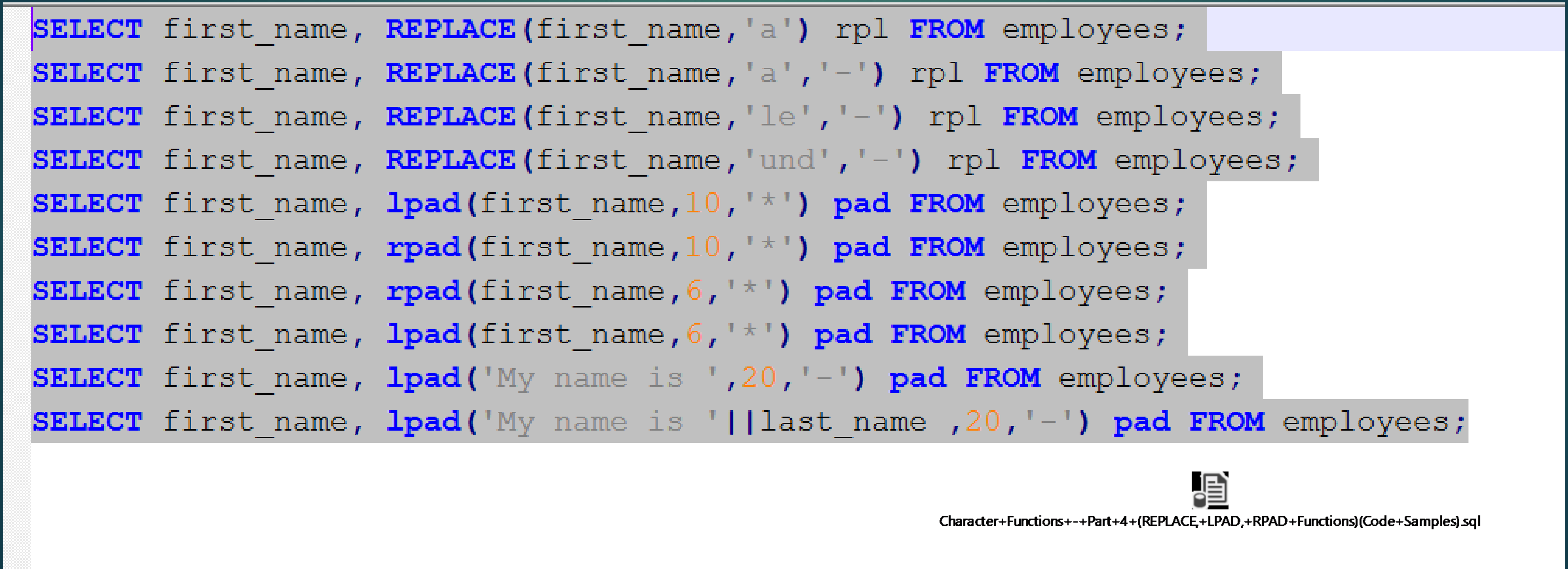


```
SELECT TRIM ('     My Name is Adam   ') trimmed_text from dual;
SELECT TRIM (' ' FROM '     My Name is Adam   ') trimmed_text from dual;
SELECT TRIM (BOTH ' ' FROM '     My Name is Adam   ') trimmed_text from dual;
SELECT TRIM (LEADING ' ' FROM '     My Name is Adam   ') trimmed_text from dual;
SELECT TRIM (TRAILING ' ' FROM '     My Name is Adam   ') trimmed_text from dual;
SELECT TRIM (TRAILING 'm' FROM '     my Name is Adam   ') trimmed_text from dual;
SELECT TRIM (TRAILING 'm' FROM 'my Name is Adam') trimmed_text from dual;
SELECT TRIM (TRAILING 'm' FROM 'my Name is Adammmmm') trimmed_text from dual;
SELECT TRIM (LEADING 'm' FROM 'my Name is Adam') trimmed_text from dual;
SELECT TRIM (BOTH 'm' FROM 'my Name is Adam') trimmed_text from dual;
SELECT TRIM ('m' FROM 'my Name is Adam') trimmed_text from dual;
SELECT TRIM ('m' FROM 'my Name is Ada') trimmed_text from dual;
SELECT TRIM (TRAILING 'm' FROM 'my Name is Ada') trimmed_text from dual;
SELECT TRIM (TRAILING 'my' FROM 'my Name is Ada') trimmed_text from dual;

SELECT RTRIM ('  my Name is Adam  ') r_trimmed_text from dual;
SELECT LTRIM ('  my Name is Adam  ') l_trimmed_text from dual;
SELECT LTRIM ('my Name is Adam', 'my') l_trimmed_text from dual;
SELECT RTRIM ('my Name is Adam', 'my') r_trimmed_text from dual;
SELECT RTRIM ('my Name is Adammmm', 'my') r_trimmed_text from dual;
SELECT LTRIM ('www.yourwebsite.com', 'w.') l_trimmed_text from dual;
SELECT RTRIM(LTRIM('www.yourwebsitename.com', 'w.'),'.com') trimmed_text from dual;
SELECT ltrim('1237982434www.yourwebsitename.com', '0123456789') trimmed_text from dual;
```

Character+Functions+-+Part+3+(TRIM,+LTRIM,+RTRIM+Functions)(Code+Samples).sql

By Eslam Khder

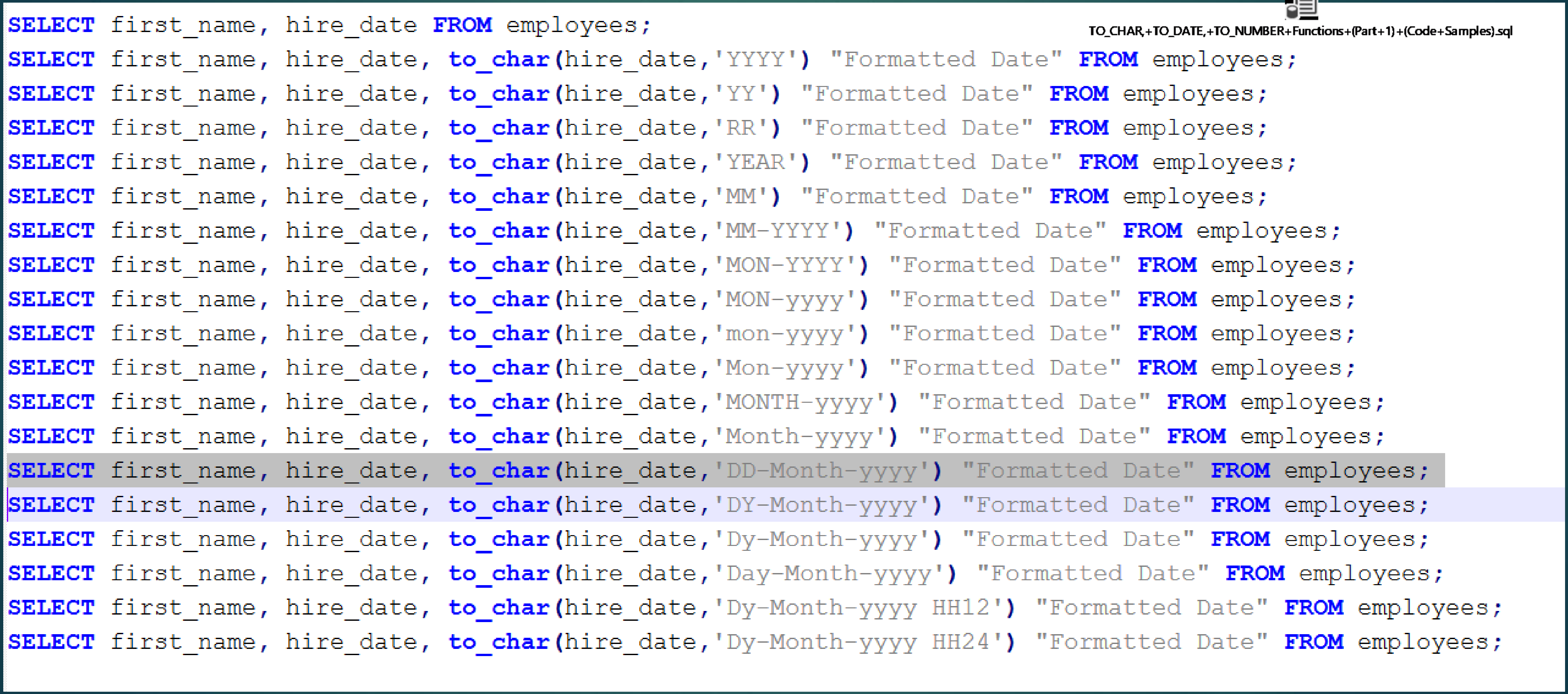
REPLACE LPAD RPAD Functions



```sql
SELECT first_name, REPLACE(first_name,'a') rpl FROM employees;
SELECT first_name, REPLACE(first_name,'a','-') rpl FROM employees;
SELECT first_name, REPLACE(first_name,'le','-') rpl FROM employees;
SELECT first_name, REPLACE(first_name,'und','-') rpl FROM employees;
SELECT first_name, lpad(first_name,10,'*') pad FROM employees;
SELECT first_name, rpad(first_name,10,'*') pad FROM employees;
SELECT first_name, rpad(first_name,6,'*') pad FROM employees;
SELECT first_name, lpad(first_name,6,'*') pad FROM employees;
SELECT first_name, lpad('My name is ',20,'-') pad FROM employees;
SELECT first_name, lpad('My name is '||last_name ,20,'-') pad FROM employees;
```

Character+Functions+-+Part+4+(REPLACE,+LPAD,+RPAD+Functions)(Code+Samples).sql

TO_CHAR



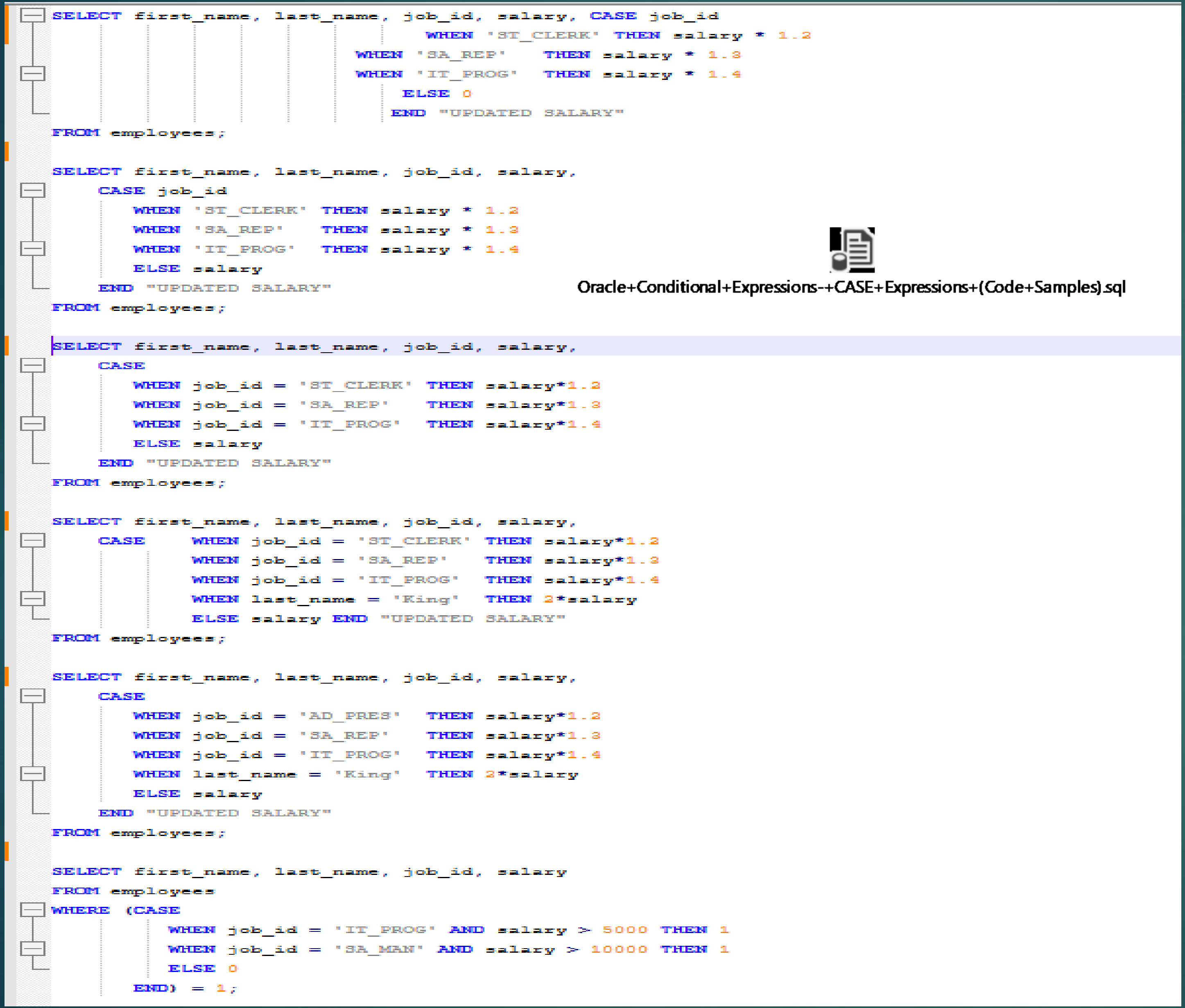
TO_CHAR,+TO_DATE,+TO_NUMBER+Functions+(Part+1)+(Code+Samples).sql

```sql
SELECT first_name, hire_date FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'YYYY') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'YY') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'RR') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'YEAR') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'MM') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'MM-YYYY') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'MON-YYYY') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'MON-yyyy') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'mon-yyyy') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'Mon-yyyy') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'MONTH-yyyy') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'Month-yyyy') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'DD-Month-yyyy') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'DY-Month-yyyy') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'Dy-Month-yyyy') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'Day-Month-yyyy') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'Dy-Month-yyyy HH12') "Formatted Date" FROM employees;
SELECT first_name, hire_date, to_char(hire_date,'Dy-Month-yyyy HH24') "Formatted Date" FROM employees;
```

By Eslam Khder

Oracle Conditional+Expressions CASE Expressions

EraaSoft

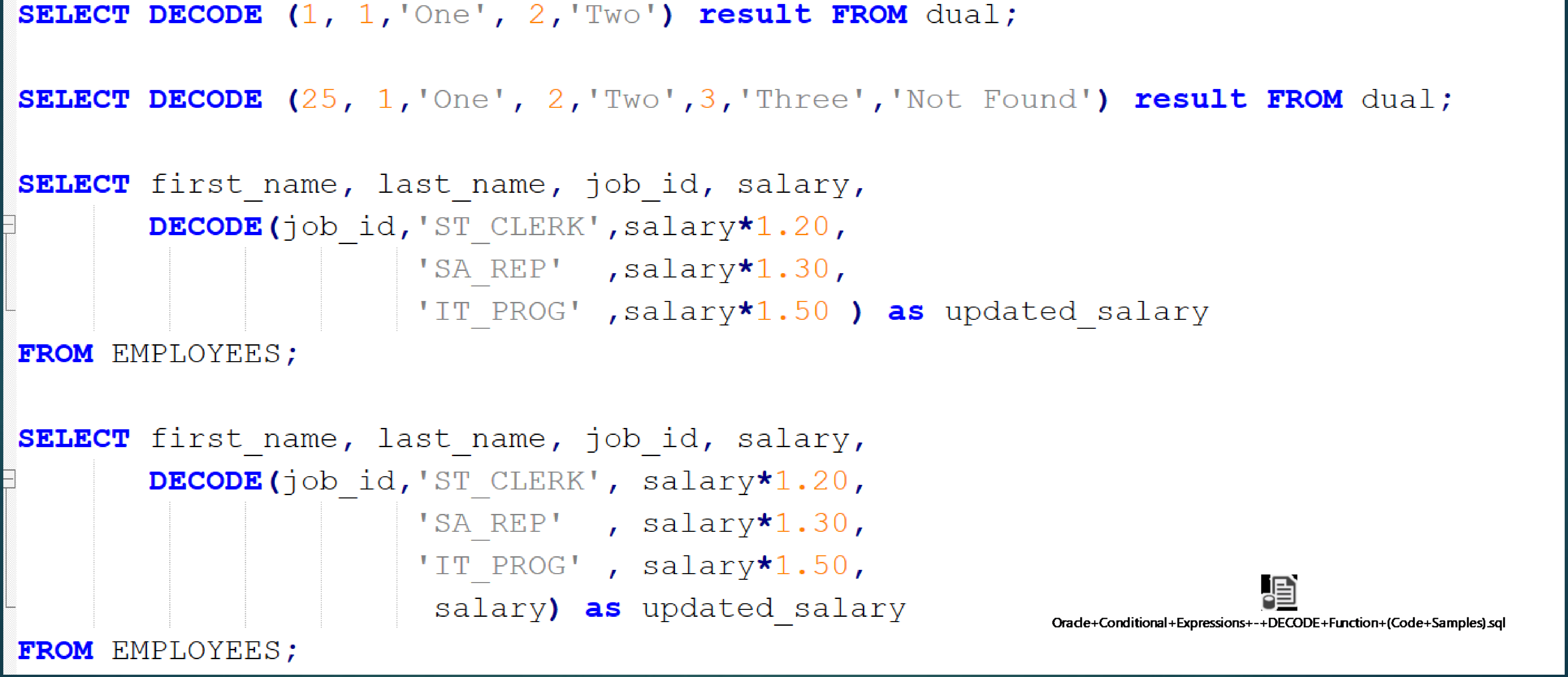


```
SELECT first_name, last_name, job_id, salary, CASE job_id
                                 WHEN 'ST_CLERK' THEN salary * 1.2
                           WHEN 'SA_REP'    THEN salary * 1.3
                           WHEN 'IT_PROG'   THEN salary * 1.4
                               ELSE 0
                              END "UPDATED SALARY"
FROM employees;

SELECT first_name, last_name, job_id, salary,
    CASE job_id
       WHEN 'ST_CLERK' THEN salary * 1.2
       WHEN 'SA_REP'   THEN salary * 1.3
       WHEN 'IT_PROG'  THEN salary * 1.4
       ELSE salary
    END "UPDATED SALARY"
FROM employees;

SELECT first_name, last_name, job_id, salary,
    CASE
       WHEN job_id = 'ST_CLERK' THEN salary*1.2
       WHEN job_id = 'SA_REP'   THEN salary*1.3
       WHEN job_id = 'IT_PROG'  THEN salary*1.4
       ELSE salary
    END "UPDATED SALARY"
FROM employees;

SELECT first_name, last_name, job_id, salary,
    CASE    WHEN job_id = 'ST_CLERK' THEN salary*1.2
            WHEN job_id = 'SA_REP'   THEN salary*1.3
            WHEN job_id = 'IT_PROG'  THEN salary*1.4
            WHEN last_name = 'King'  THEN 2*salary
            ELSE salary END "UPDATED SALARY"
FROM employees;

SELECT first_name, last_name, job_id, salary,
    CASE
       WHEN job_id = 'AD_PRES'  THEN salary*1.2
       WHEN job_id = 'SA_REP'   THEN salary*1.3
       WHEN job_id = 'IT_PROG'  THEN salary*1.4
       WHEN last_name = 'King'  THEN 2*salary
       ELSE salary
    END "UPDATED SALARY"
FROM employees;

SELECT first_name, last_name, job_id, salary
FROM employees
WHERE (CASE
          WHEN job_id = 'IT_PROG' AND salary > 5000 THEN 1
          WHEN job_id = 'SA_MAN' AND salary > 10000 THEN 1
          ELSE 0
       END) = 1;
```

Oracle+Conditional+Expressions-+CASE+Expressions+(Code+Samples).sql

By Eslam Khder

Oracle Conditional Expressions DECODE Function



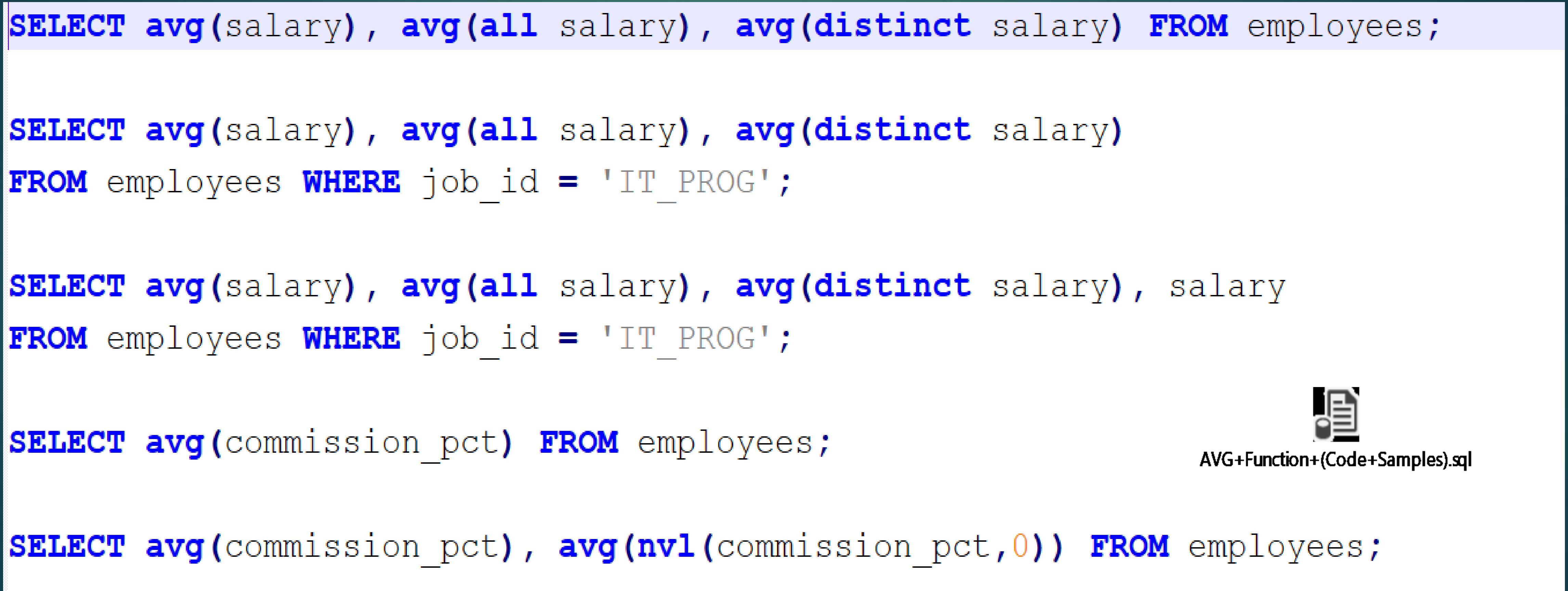
```
SELECT DECODE (1, 1,'One', 2,'Two') result FROM dual;

SELECT DECODE (25, 1,'One', 2,'Two',3,'Three','Not Found') result FROM dual;

SELECT first_name, last_name, job_id, salary,
       DECODE(job_id,'ST_CLERK',salary*1.20,
                     'SA_REP'  ,salary*1.30,
                     'IT_PROG' ,salary*1.50 ) as updated_salary
FROM EMPLOYEES;

SELECT first_name, last_name, job_id, salary,
       DECODE(job_id,'ST_CLERK', salary*1.20,
                     'SA_REP'  , salary*1.30,
                     'IT_PROG' , salary*1.50,
                      salary) as updated_salary
FROM EMPLOYEES;
```

Oracle+Conditional+Expressions+-+DECODE+Function+(Code+Samples).sql

By Eslam Khder

EraaSoft

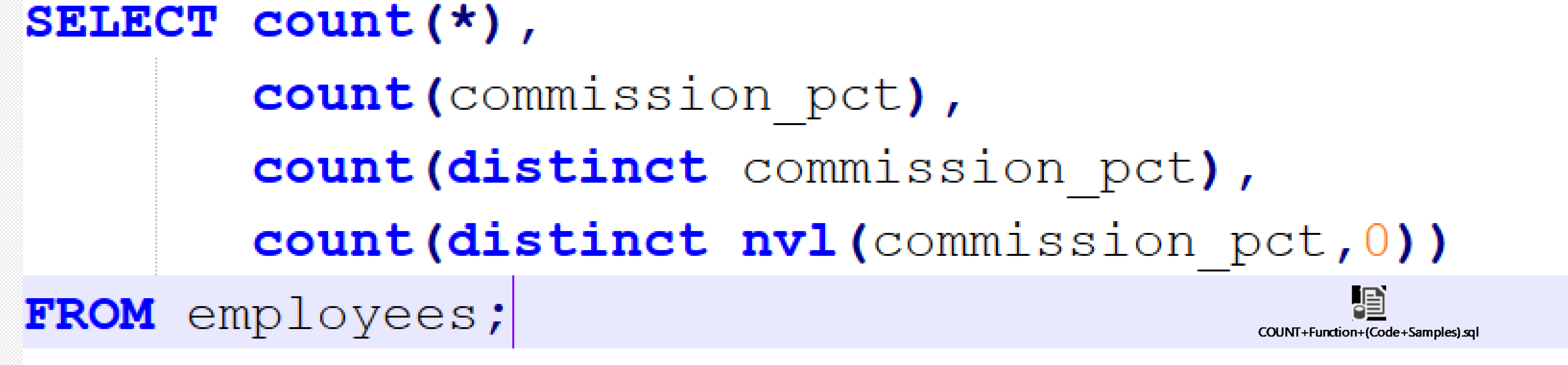
AVG Function



```
SELECT avg(salary), avg(all salary), avg(distinct salary) FROM employees;

SELECT avg(salary), avg(all salary), avg(distinct salary)
FROM employees WHERE job_id = 'IT_PROG';

SELECT avg(salary), avg(all salary), avg(distinct salary), salary
FROM employees WHERE job_id = 'IT_PROG';

SELECT avg(commission_pct) FROM employees;

SELECT avg(commission_pct), avg(nvl(commission_pct,0)) FROM employees;
```

AVG+Function+(Code+Samples).sql

COUNT Function

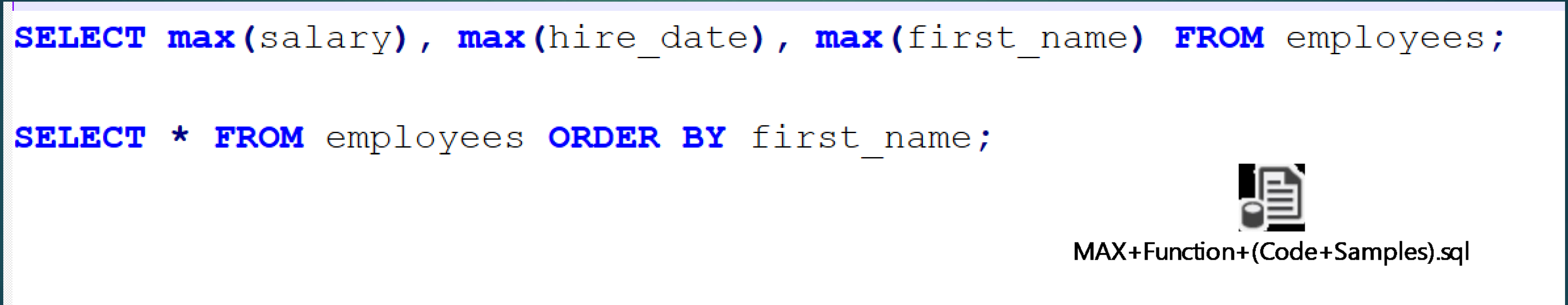


```sql
SELECT count(*),
       count(commission_pct),
       count(distinct commission_pct),
       count(distinct nvl(commission_pct,0))
FROM employees;
```

COUNT+Function+(Code+Samples).sql

By Eslam Khder

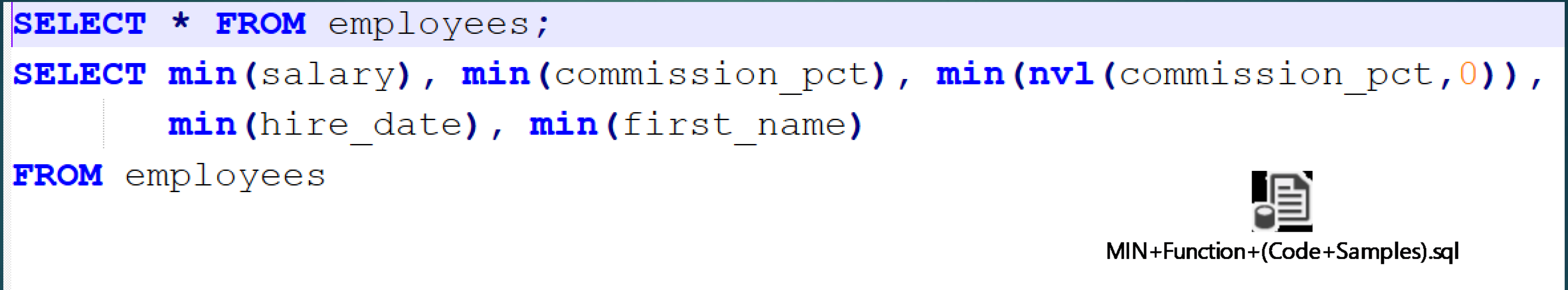
MAX Function



```
SELECT max(salary), max(hire_date), max(first_name) FROM employees;

SELECT * FROM employees ORDER BY first_name;
```

MAX+Function+(Code+Samples).sql

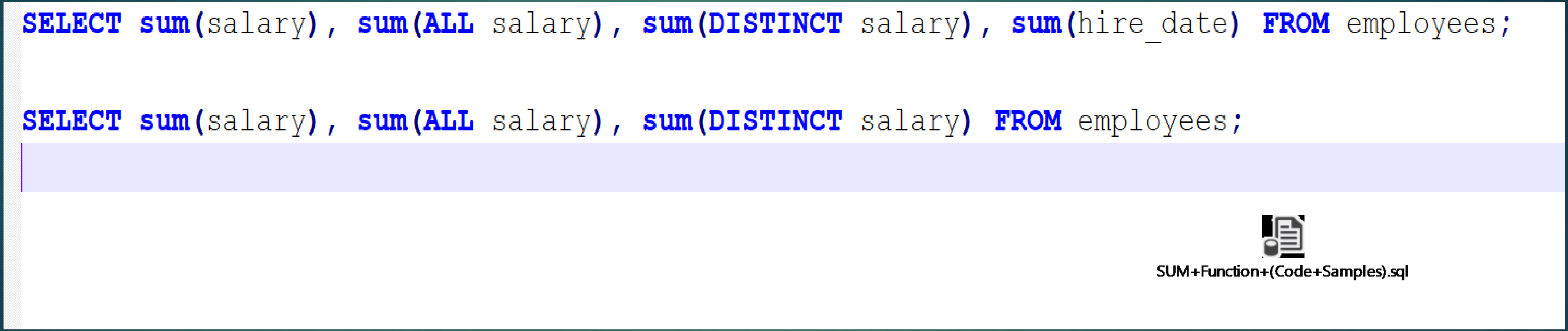
MIN Function



```
SELECT * FROM employees;
SELECT min(salary), min(commission_pct), min(nvl(commission_pct,0)),
       min(hire_date), min(first_name)
FROM employees
```

MIN+Function+(Code+Samples).sql

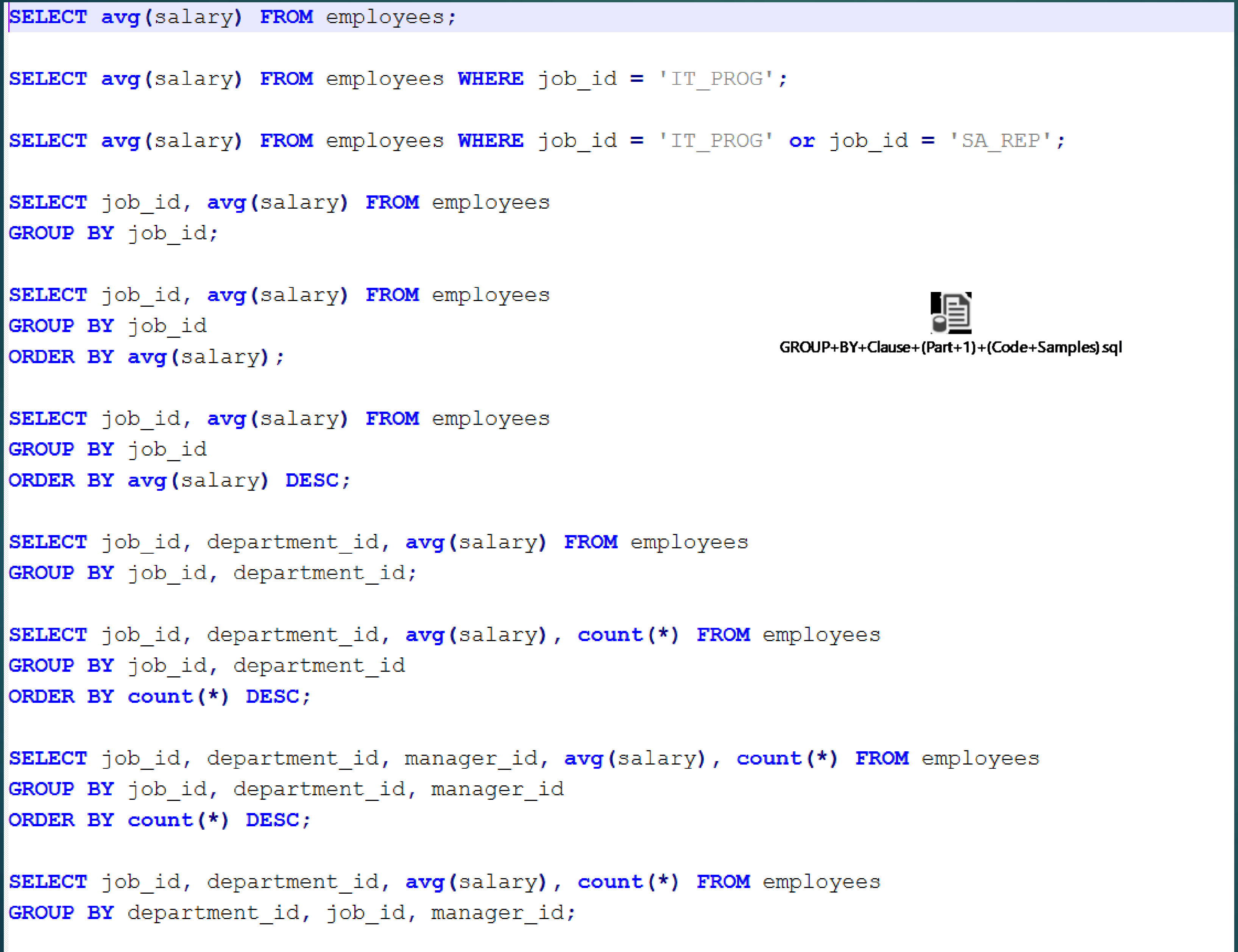
SUM Function



```
SELECT sum(salary), sum(ALL salary), sum(DISTINCT salary), sum(hire_date) FROM employees;

SELECT sum(salary), sum(ALL salary), sum(DISTINCT salary) FROM employees;
```

SUM+Function+(Code+Samples).sql

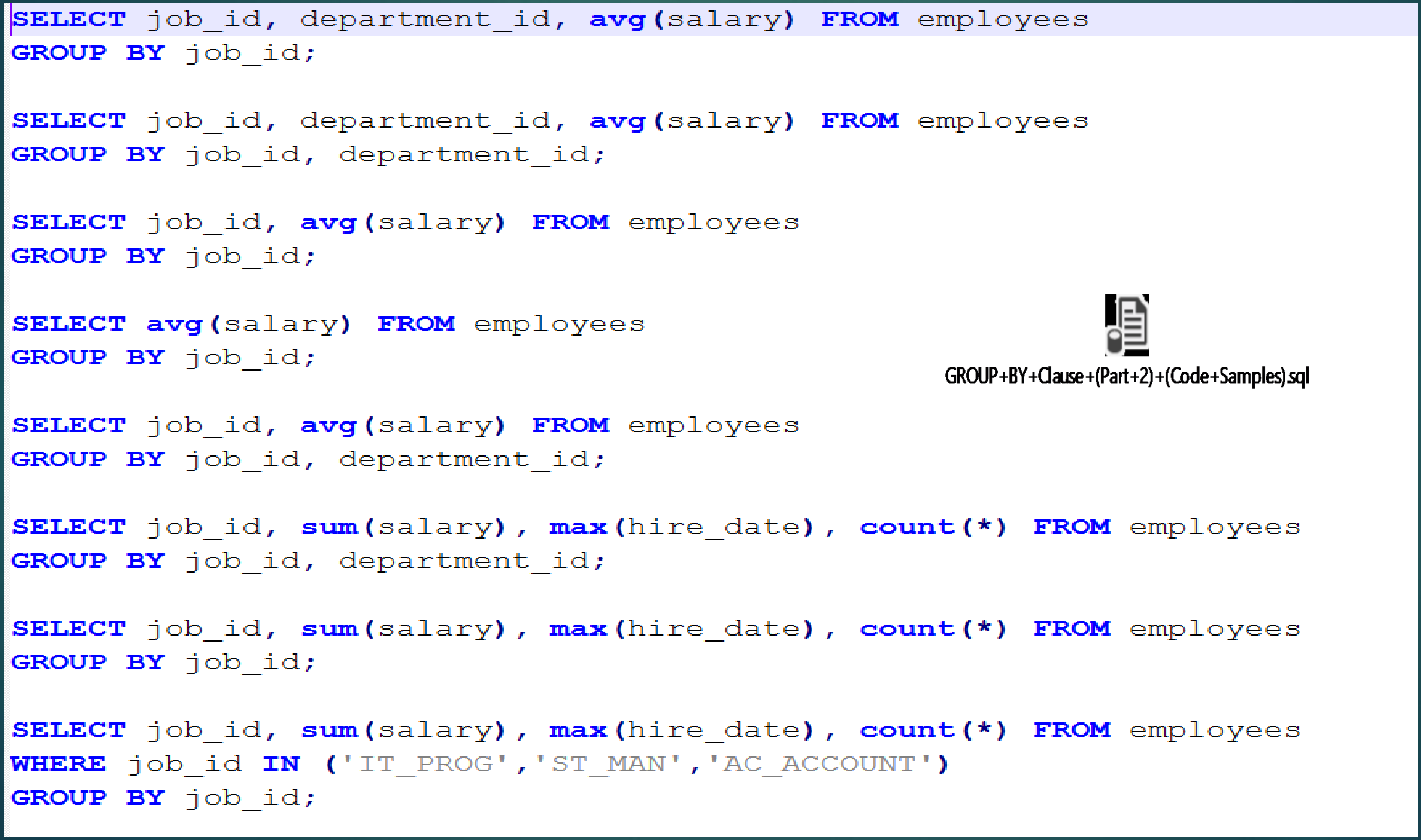
GROUP BY Clause



```
SELECT avg(salary) FROM employees;

SELECT avg(salary) FROM employees WHERE job_id = 'IT_PROG';

SELECT avg(salary) FROM employees WHERE job_id = 'IT_PROG' or job_id = 'SA_REP';

SELECT job_id, avg(salary) FROM employees
GROUP BY job_id;

SELECT job_id, avg(salary) FROM employees
GROUP BY job_id
ORDER BY avg(salary);

SELECT job_id, avg(salary) FROM employees
GROUP BY job_id
ORDER BY avg(salary) DESC;

SELECT job_id, department_id, avg(salary) FROM employees
GROUP BY job_id, department_id;

SELECT job_id, department_id, avg(salary), count(*) FROM employees
GROUP BY job_id, department_id
ORDER BY count(*) DESC;

SELECT job_id, department_id, manager_id, avg(salary), count(*) FROM employees
GROUP BY job_id, department_id, manager_id
ORDER BY count(*) DESC;

SELECT job_id, department_id, avg(salary), count(*) FROM employees
GROUP BY department_id, job_id, manager_id;
```

GROUP+BY+Clause+(Part+1)+(Code+Samples).sql

GROUP BY Clause

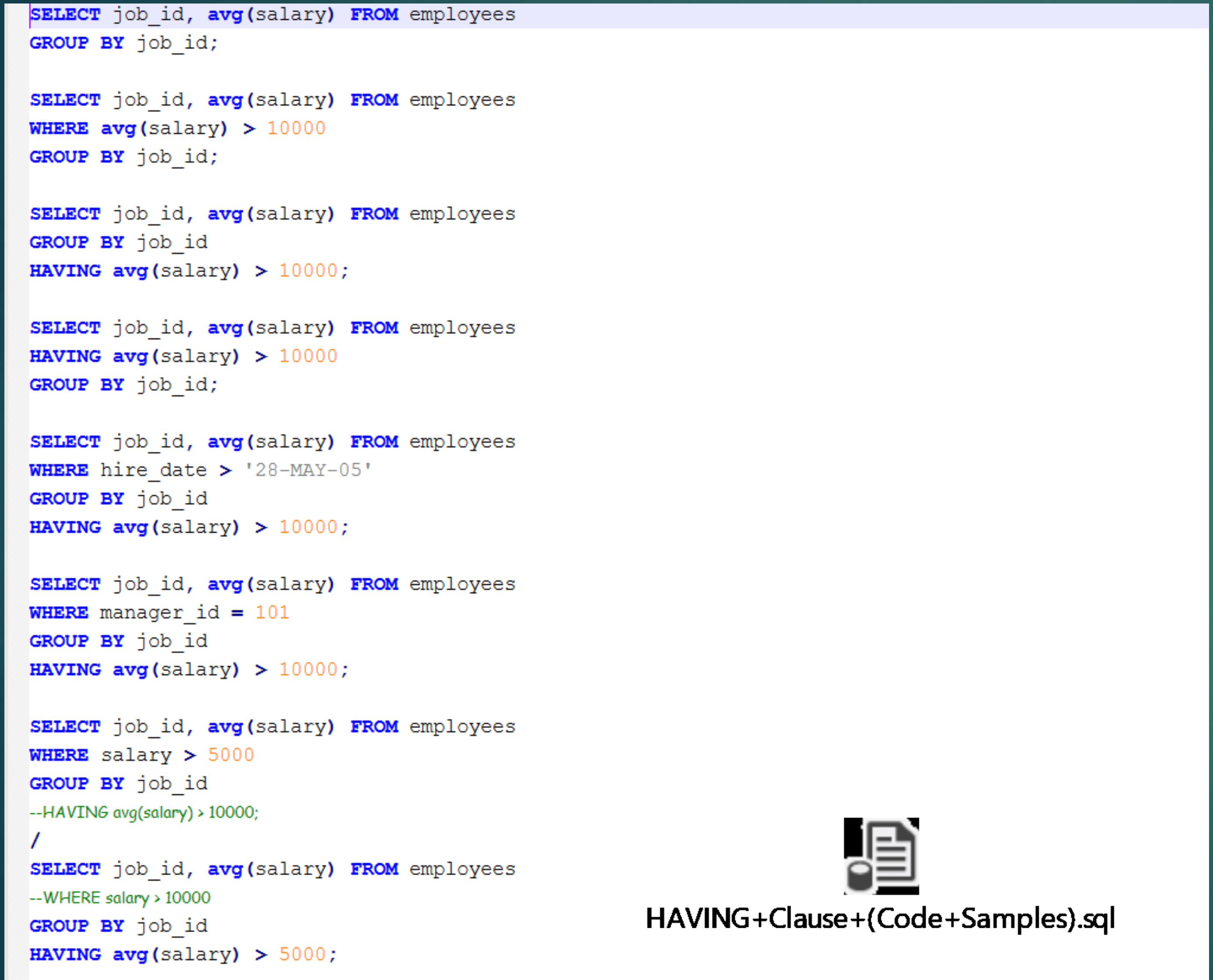


```
SELECT job_id, department_id, avg(salary) FROM employees
GROUP BY job_id;

SELECT job_id, department_id, avg(salary) FROM employees
GROUP BY job_id, department_id;

SELECT job_id, avg(salary) FROM employees
GROUP BY job_id;

SELECT avg(salary) FROM employees
GROUP BY job_id;

SELECT job_id, avg(salary) FROM employees
GROUP BY job_id, department_id;

SELECT job_id, sum(salary), max(hire_date), count(*) FROM employees
GROUP BY job_id, department_id;

SELECT job_id, sum(salary), max(hire_date), count(*) FROM employees
GROUP BY job_id;

SELECT job_id, sum(salary), max(hire_date), count(*) FROM employees
WHERE job_id IN ('IT_PROG','ST_MAN','AC_ACCOUNT')
GROUP BY job_id;
```

GROUP+BY+Clause+(Part+2)+(Code+Samples).sql

By Eslam Khder

HAVING Clause



```sql
SELECT job_id, avg(salary) FROM employees
GROUP BY job_id;

SELECT job_id, avg(salary) FROM employees
WHERE avg(salary) > 10000
GROUP BY job_id;

SELECT job_id, avg(salary) FROM employees
GROUP BY job_id
HAVING avg(salary) > 10000;

SELECT job_id, avg(salary) FROM employees
HAVING avg(salary) > 10000
GROUP BY job_id;

SELECT job_id, avg(salary) FROM employees
WHERE hire_date > '28-MAY-05'
GROUP BY job_id
HAVING avg(salary) > 10000;

SELECT job_id, avg(salary) FROM employees
WHERE manager_id = 101
GROUP BY job_id
HAVING avg(salary) > 10000;

SELECT job_id, avg(salary) FROM employees
WHERE salary > 5000
GROUP BY job_id
--HAVING avg(salary) > 10000;
/
SELECT job_id, avg(salary) FROM employees
--WHERE salary > 10000
GROUP BY job_id
HAVING avg(salary) > 5000;
```

HAVING+Clause+(Code+Samples).sql

Join with the USING Clause

Natural Join

Join with the USING Clause

Inner Join

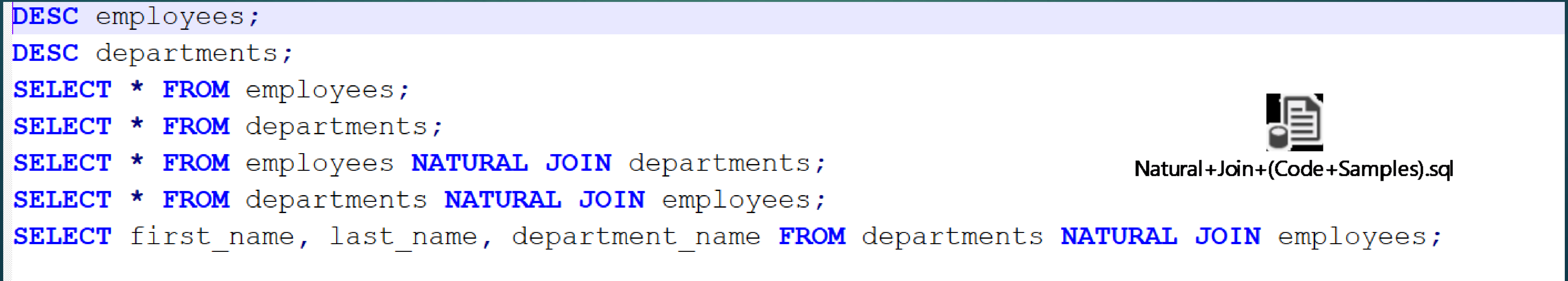
LEFT JOIN (or LEFT OUTER JOIN)

RIGHT JOIN (or RIGHT OUTER JOIN)

FULL JOIN (or FULL OUTER JOIN)

Natural Join

A NATURAL JOIN in Oracle Database is a type of join that automatically joins tables based on columns with the same name and compatible data types in .both tables



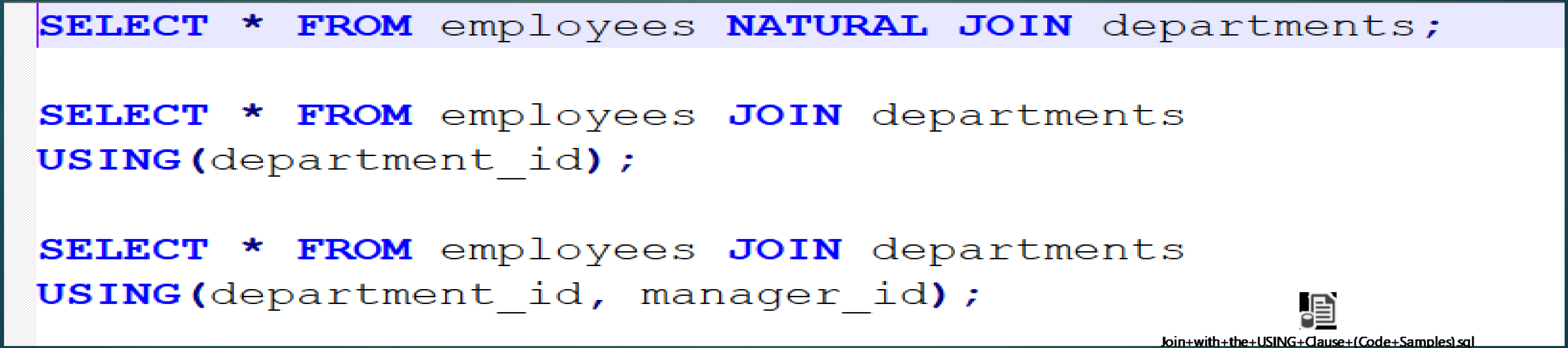
```sql
DESC employees;
DESC departments;
SELECT * FROM employees;
SELECT * FROM departments;
SELECT * FROM employees NATURAL JOIN departments;
SELECT * FROM departments NATURAL JOIN employees;
SELECT first_name, last_name, department_name FROM departments NATURAL JOIN employees;
```

Natural+Join+(Code+Samples).sql

By Eslam Khder

EraaSoft

Join with the USING Clause

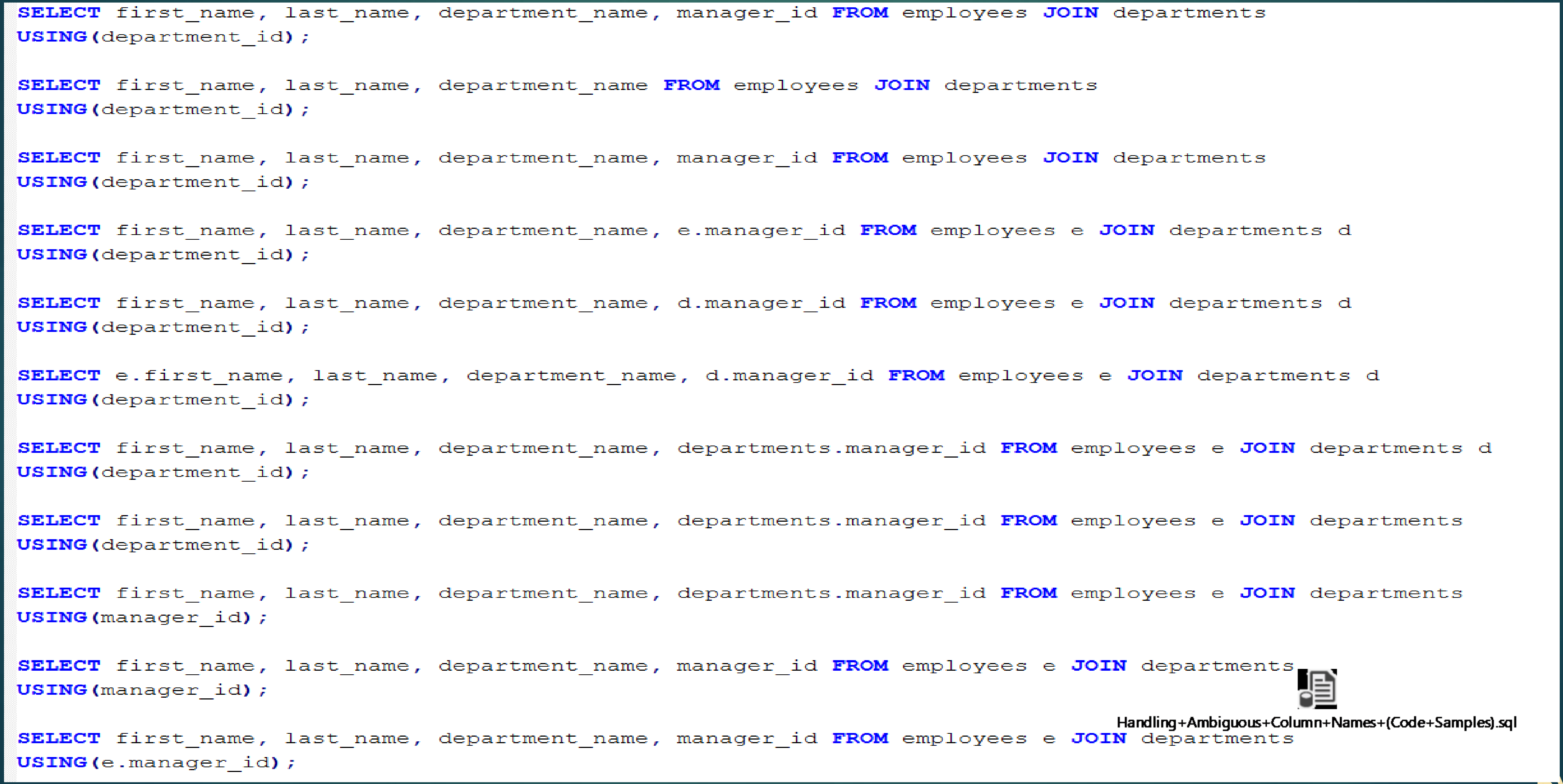


```
SELECT * FROM employees NATURAL JOIN departments;

SELECT * FROM employees JOIN departments
USING(department_id);

SELECT * FROM employees JOIN departments
USING(department_id, manager_id);
```

Join+with+the+USING+Clause+(Code+Samples).sql

Handling Ambiguous Column Names

EraaSoft



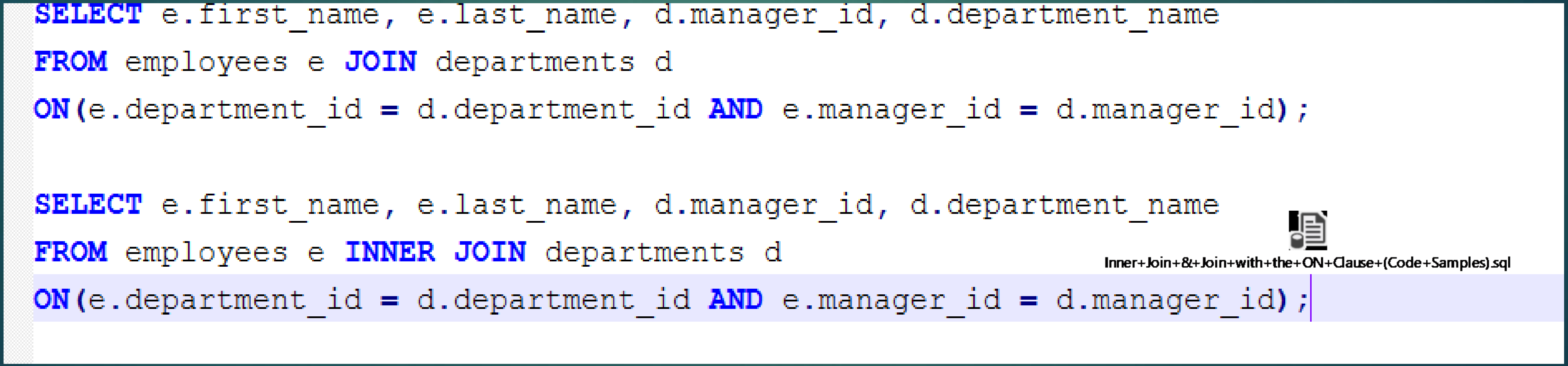
```
SELECT first_name, last_name, department_name, manager_id FROM employees JOIN departments
USING(department_id);

SELECT first_name, last_name, department_name FROM employees JOIN departments
USING(department_id);

SELECT first_name, last_name, department_name, manager_id FROM employees JOIN departments
USING(department_id);

SELECT first_name, last_name, department_name, e.manager_id FROM employees e JOIN departments d
USING(department_id);

SELECT first_name, last_name, department_name, d.manager_id FROM employees e JOIN departments d
USING(department_id);

SELECT e.first_name, last_name, department_name, d.manager_id FROM employees e JOIN departments d
USING(department_id);

SELECT first_name, last_name, department_name, departments.manager_id FROM employees e JOIN departments d
USING(department_id);

SELECT first_name, last_name, department_name, departments.manager_id FROM employees e JOIN departments
USING(department_id);

SELECT first_name, last_name, department_name, departments.manager_id FROM employees e JOIN departments
USING(manager_id);

SELECT first_name, last_name, department_name, manager_id FROM employees e JOIN departments
USING(manager_id);

SELECT first_name, last_name, department_name, manager_id FROM employees e JOIN departments
USING(e.manager_id);
```

Handling+Ambiguous+Column+Names+(Code+Samples).sql

By Eslam Khder

Inner Join Join with the ON Clause

An INNER JOIN returns rows when there is a match in both
.tables. It is the most common type of join

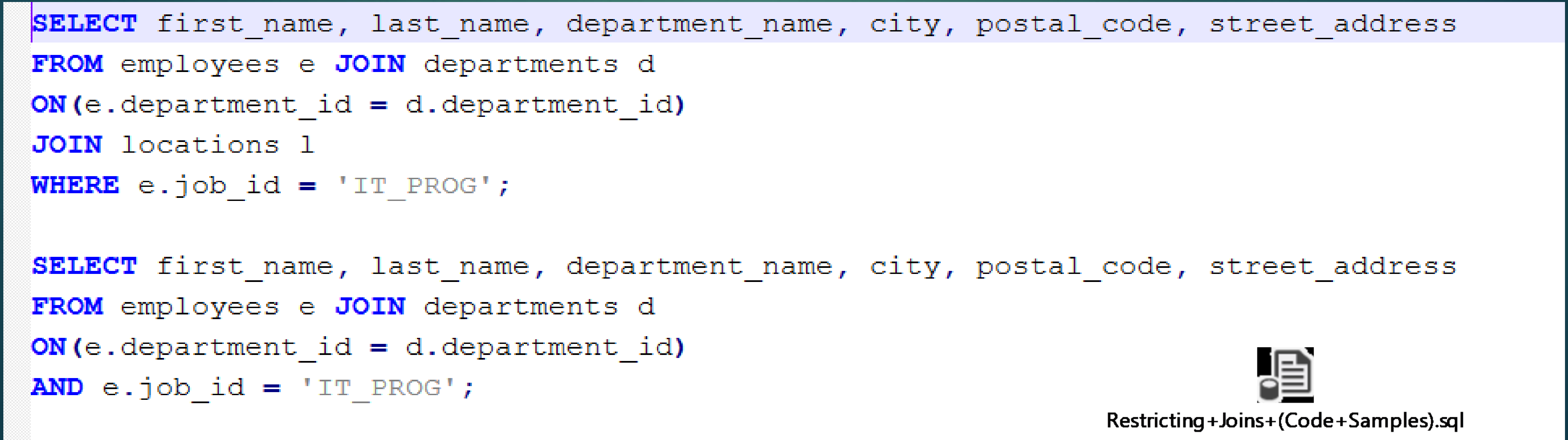


```sql
SELECT e.first_name, e.last_name, d.manager_id, d.department_name
FROM employees e JOIN departments d
ON(e.department_id = d.department_id AND e.manager_id = d.manager_id);

SELECT e.first_name, e.last_name, d.manager_id, d.department_name
FROM employees e INNER JOIN departments d
ON(e.department_id = d.department_id AND e.manager_id = d.manager_id);
```

Inner+Join+&+Join+with+the+ON+Clause+(Code+Samples).sql

By Eslam Khder

Restricting Joins



```
SELECT first_name, last_name, department_name, city, postal_code, street_address
FROM employees e JOIN departments d
ON(e.department_id = d.department_id)
JOIN locations l
WHERE e.job_id = 'IT_PROG';

SELECT first_name, last_name, department_name, city, postal_code, street_address
FROM employees e JOIN departments d
ON(e.department_id = d.department_id)
AND e.job_id = 'IT_PROG';
```

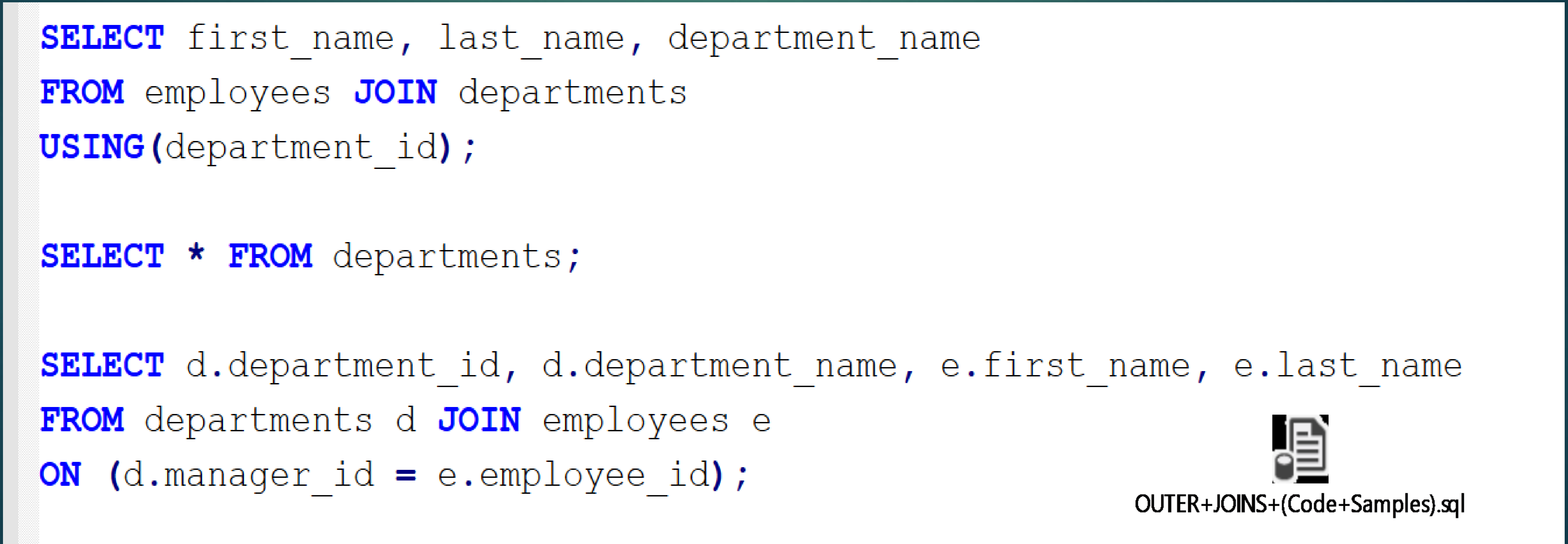
Restricting+Joins+(Code+Samples).sql

Outer Join

LEFT JOIN (or LEFT OUTER JOIN)

RIGHT JOIN (or RIGHT OUTER JOIN)

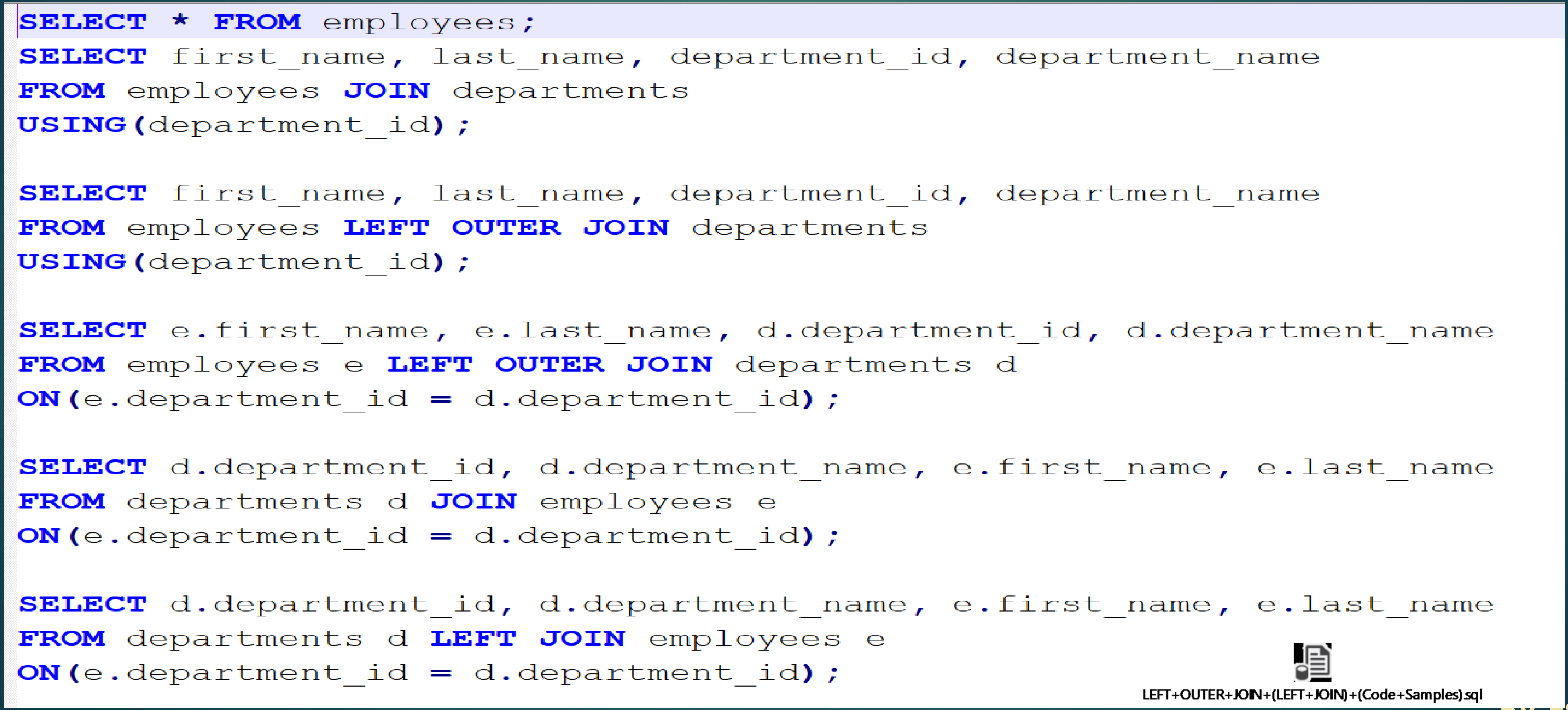
FULL JOIN (or FULL OUTER JOIN)



```
SELECT first_name, last_name, department_name
FROM employees JOIN departments
USING(department_id);

SELECT * FROM departments;

SELECT d.department_id, d.department_name, e.first_name, e.last_name
FROM departments d JOIN employees e
ON (d.manager_id = e.employee_id);
```

OUTER+JOINS+(Code+Samples).sql

LEFT JOIN (or LEFT OUTER JOIN)



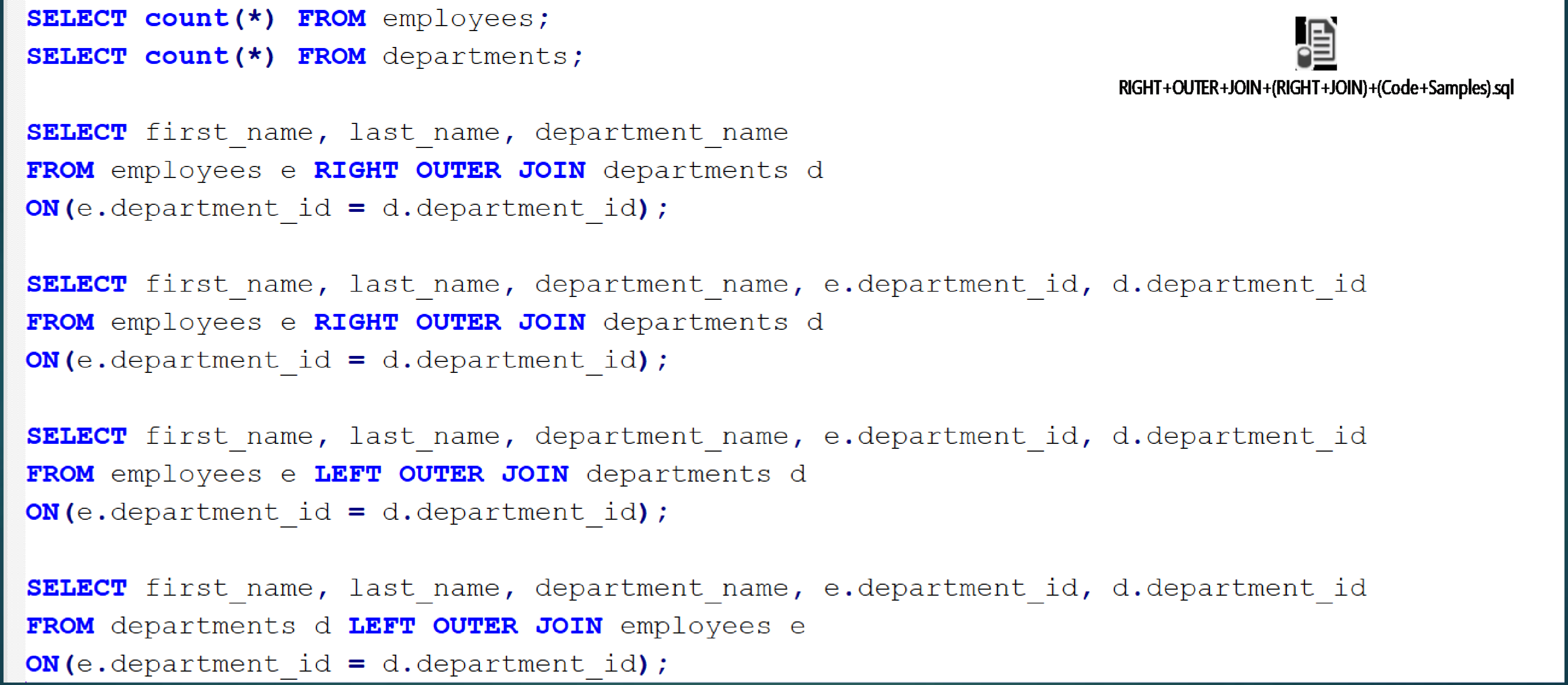
```
SELECT * FROM employees;
SELECT first_name, last_name, department_id, department_name
FROM employees JOIN departments
USING(department_id);

SELECT first_name, last_name, department_id, department_name
FROM employees LEFT OUTER JOIN departments
USING(department_id);

SELECT e.first_name, e.last_name, d.department_id, d.department_name
FROM employees e LEFT OUTER JOIN departments d
ON(e.department_id = d.department_id);

SELECT d.department_id, d.department_name, e.first_name, e.last_name
FROM departments d JOIN employees e
ON(e.department_id = d.department_id);

SELECT d.department_id, d.department_name, e.first_name, e.last_name
FROM departments d LEFT JOIN employees e
ON(e.department_id = d.department_id);
```

LEFT+OUTER+JOIN+(LEFT+JOIN)+(Code+Samples).sql

By Eslam Khder

EraaSoft

RIGHT JOIN (or RIGHT OUTER JOIN)

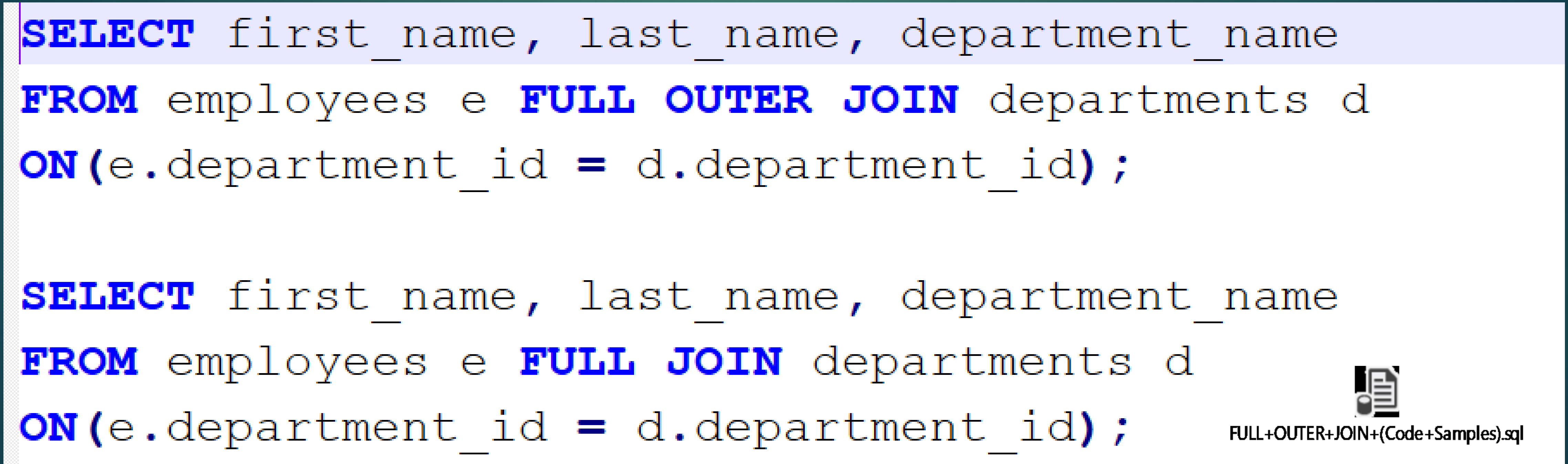


```
SELECT count(*) FROM employees;
SELECT count(*) FROM departments;

SELECT first_name, last_name, department_name
FROM employees e RIGHT OUTER JOIN departments d
ON(e.department_id = d.department_id);

SELECT first_name, last_name, department_name, e.department_id, d.department_id
FROM employees e RIGHT OUTER JOIN departments d
ON(e.department_id = d.department_id);

SELECT first_name, last_name, department_name, e.department_id, d.department_id
FROM employees e LEFT OUTER JOIN departments d
ON(e.department_id = d.department_id);

SELECT first_name, last_name, department_name, e.department_id, d.department_id
FROM departments d LEFT OUTER JOIN employees e
ON(e.department_id = d.department_id);
```

RIGHT+OUTER+JOIN+(RIGHT+JOIN)+(Code+Samples).sql

By Eslam Khder

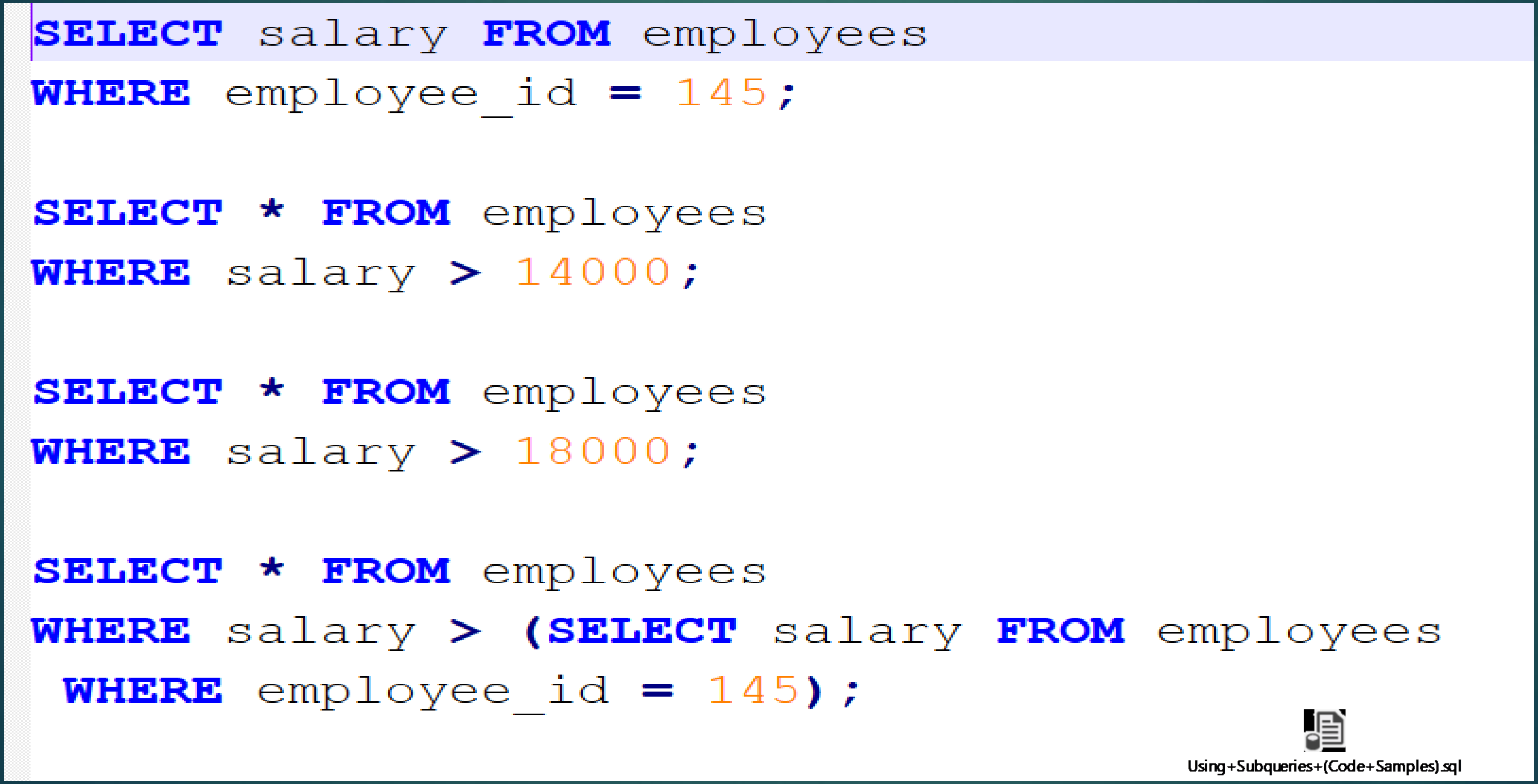
FULL JOIN (or FULL OUTER JOIN)



```
SELECT first_name, last_name, department_name
FROM employees e FULL OUTER JOIN departments d
ON(e.department_id = d.department_id);

SELECT first_name, last_name, department_name
FROM employees e FULL JOIN departments d
ON(e.department_id = d.department_id);
```

FULL+OUTER+JOIN+(Code+Samples).sql

Using Subqueries

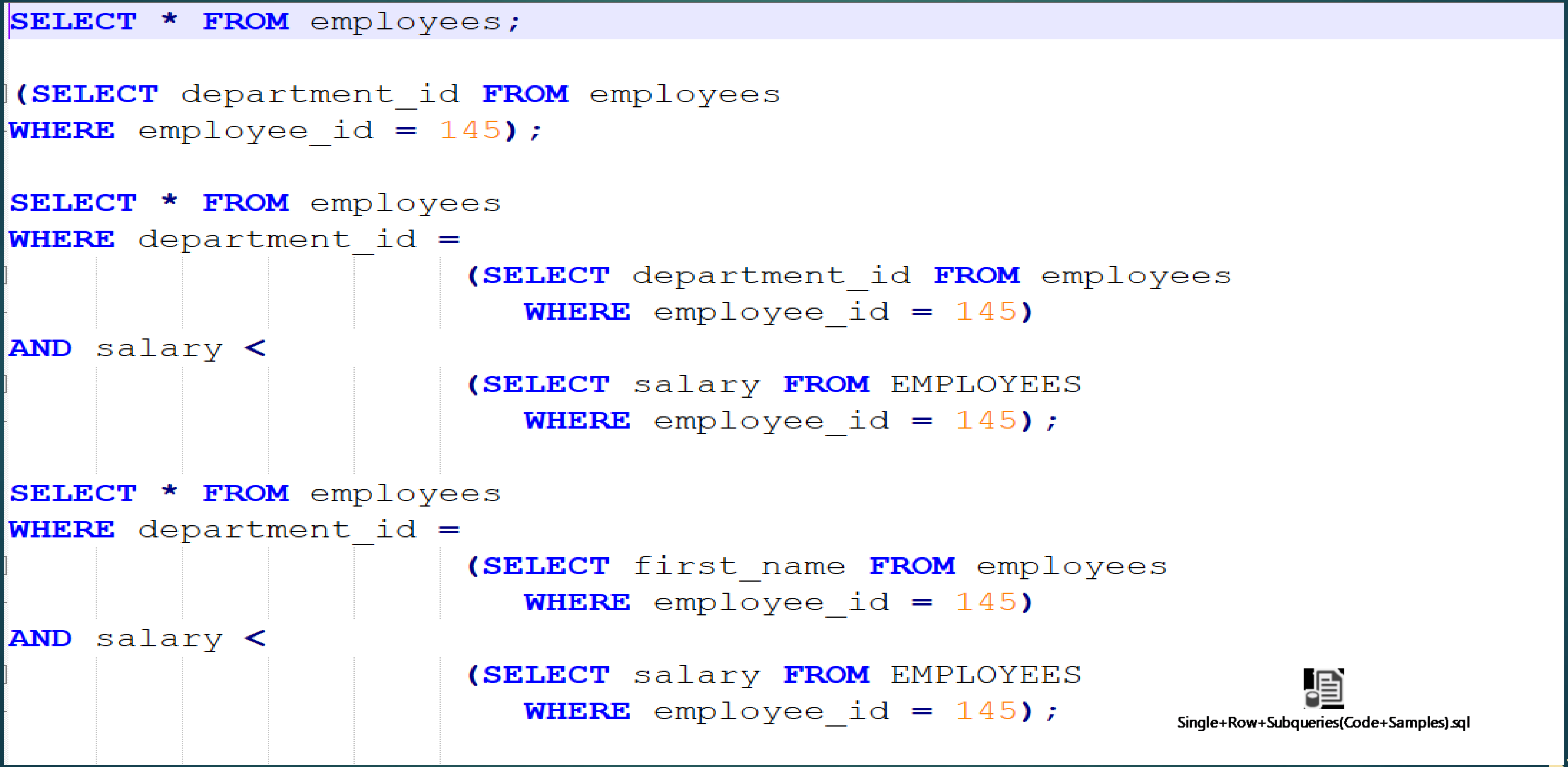


```
SELECT salary FROM employees
WHERE employee_id = 145;

SELECT * FROM employees
WHERE salary > 14000;

SELECT * FROM employees
WHERE salary > 18000;

SELECT * FROM employees
WHERE salary > (SELECT salary FROM employees
 WHERE employee_id = 145);
```

Using+Subqueries+(Code+Samples).sql

By Eslam Khder

Single Row Subqueries

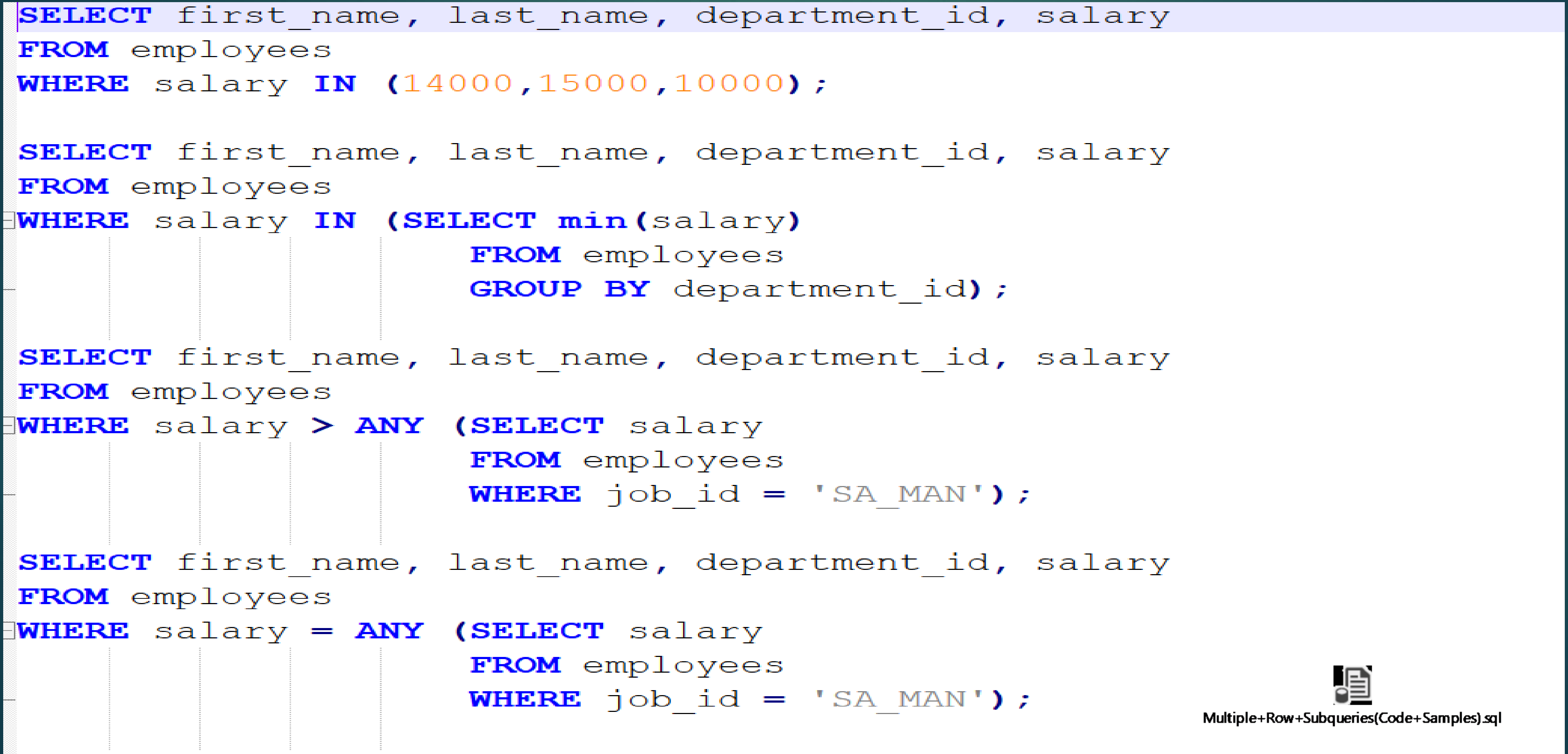


```sql
SELECT * FROM employees;

(SELECT department_id FROM employees
WHERE employee_id = 145);

SELECT * FROM employees
WHERE department_id =
                      (SELECT department_id FROM employees
                        WHERE employee_id = 145)
AND salary <
                      (SELECT salary FROM EMPLOYEES
                        WHERE employee_id = 145);

SELECT * FROM employees
WHERE department_id =
                      (SELECT first_name FROM employees
                        WHERE employee_id = 145)
AND salary <
                      (SELECT salary FROM EMPLOYEES
                        WHERE employee_id = 145);
```

Single+Row+Subqueries(Code+Samples).sql

By Eslam Khder

Multiple Row Subqueries



```sql
SELECT first_name, last_name, department_id, salary
FROM employees
WHERE salary IN (14000,15000,10000);

SELECT first_name, last_name, department_id, salary
FROM employees
WHERE salary IN (SELECT min(salary)
                    FROM employees
                    GROUP BY department_id);

SELECT first_name, last_name, department_id, salary
FROM employees
WHERE salary > ANY (SELECT salary
                    FROM employees
                    WHERE job_id = 'SA_MAN');

SELECT first_name, last_name, department_id, salary
FROM employees
WHERE salary = ANY (SELECT salary
                    FROM employees
                    WHERE job_id = 'SA_MAN');
```

Multiple+Row+Subqueries(Code+Samples).sql

By Eslam Khder

By Eslam Khder