datametica
GIVING DATA WINGS

RAVEN-OBJECT Billing Document

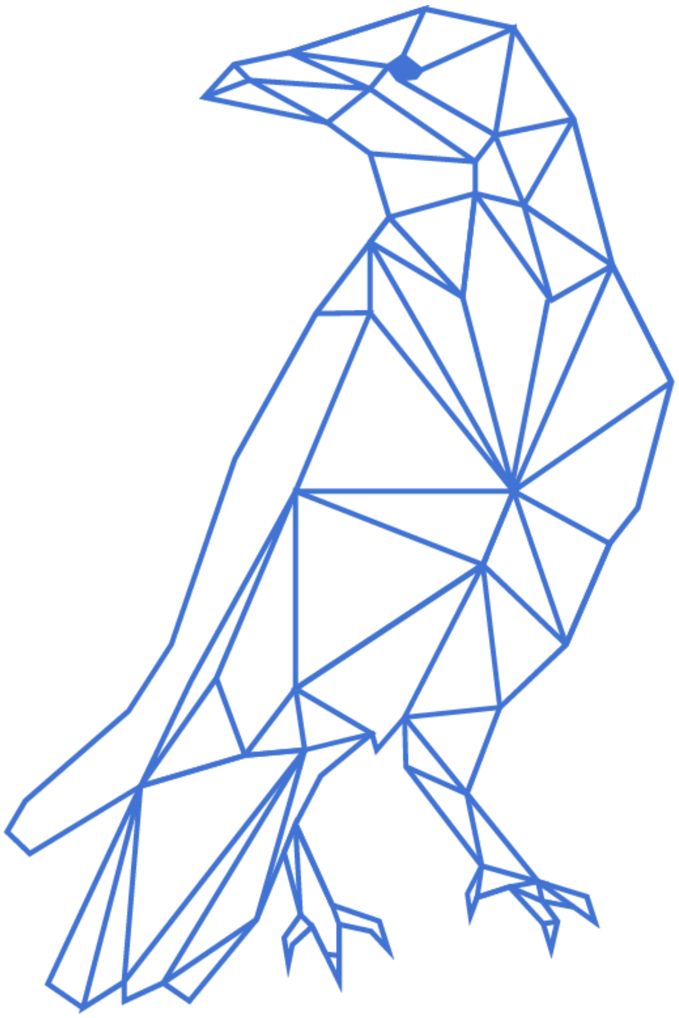

Table of Content

1. Introduction:

Every database consists of multiple DB objects, sql statements, procedural objects and scripts which have varying degrees of complexities. This includes table creation DDL which does not have business logic and is generally simple to handle.

On the other hand there exists extremely complex business logic implementations in Stored Procedures. RAVEN-Object eliminates this complexity by bringing all the database components to a single common denominator. This is done by counting the number of objects in conjunction with assigning weightage to the database objects.

The purpose of this document is to describe how RAVEN calculates the raven-objects during translation for usage and billing.

1.1. Table, views and standalone SQL

While translating a DDL or standalone SQL, RAVEN counts all the references of all tables and views. If a table/view occurs multiple times in a query, RAVEN increases the counter by 1 for every reference.

Object count is displayed in Raven Web translation page and this information is included in Translation Summary Report. It provides :

Total Objets : The total number of objects present in a batch of input DDL/standalone SQL

Translated Object : Total number of successfully translated objects

Following are few examples:

Eg.1

Select Statement -- SQL

Table or view present in the from clause will be counted as an object.

```
SELECT * FROM trimmed_salary     --   object 1
INNER JOIN department            --   object 2
ON trimmed_salary.department_id = department.department_id;
```

Eg.2

```
SELECT * FROM trimmed_salary e  --        object 1
INNER JOIN
(SELECT * FROM department    --    object 2
) d
ON e.department_id = d.department_id
```

Eg.3

```
CREATE TABLE trimmed_employee_sample          --        object 1
AS (SELECT * FROM foodmart.trimmed_employee);          --        object 2
```

Eg.4

Create View Statement

```
CREATE VIEW view1            --        object 1
AS (SELECT * FROM foodmart.trimmed_salary);    --        object 2
```

Eg.5

Insert Statement --SQL

```
INSERT INTO emp (id, name) select department_id, department_description        --        object 1
FROM foodmart.department              --        object 2
WHERE department_id = 16;
```

Eg.6

Update Statement --SQL

```
UPDATE foodmart.table1      --        object 1
FROM foodmart.table2        --        object 2
SET id = table1.id ,name = table1.name ,surname = table2.middleName
WHERE table2.id >= 6 and table1.name = table2.name;
```

1.2. Procedural objects and Scripts

Databases also provide procedural object and scripts such as scripts with Bteq, Macros, Stored Procedures, TPT etc
While translating procedural objects and scripts RAVEN bifurcates the entire code base to SQL statements and programming constructs. Object counting for DDL/SQL is explained earlier in this document.
For programming constructs following constructs are counted as 1 object.

- IF
- WHILE
- ITERATE
- LOOP
- CASE
- FOR LOOP...END FOR
- DECLARE CURSOR FOR
- IF..END IF

Following are few example to count objects for different programing constructs:

Eg.1

```
.IF ACTIVITYCOUNT = 0 THEN .Goto TableCreate;          --------   object 1

select c from ( select count(*) as c from foodmart.deptarment where databasename =
'STARSD_SL_WORK'and tablename = 'WRK_DGS_CALCULATION_06'  ----     object 2
and columnname IN( 'XYZ')) s where c=1;
```

Eg. 2

```
CREATE PROCEDURE InsertSalary ( IN in_EmployeeNo INTEGER, IN in_Gross INTEGER)
BEGIN
  INSERT INTO Salary ( EmployeeNo, Gross)  ----- Object 1
  VALUES ( :in_EmployeeNo, :in_Gross )
);
END;
```

Eg. 3

```
REPLACE PROCEDURE RETAIL_LIST_SP(IN Column_List VARCHAR(800), IN Alias_
VARCHAR(200),
OUT Aliased_Col_List VARCHAR(800))
BEGIN
        DECLARE Col_List, Cur_Val, IColumn_List VARCHAR(800) DEFAULT '';
        DECLARE iPos, Indi INT;
        SET IColumn_List = Column_List || ',';
        SET Indi = 1;
        WHILE INDEX(IColumn_List, ',') > 0          ----     Object 1
        DO
                SET iPos = INDEX(IColumn_List, ',');
        END WHILE;
        SET Aliased_Col_List = Col_List;
END;
```

1.3. Weightages for DB objects

Raven also assigns weightage to the database object.

Following are the DB objects and their weightages.

DB object	Weightage	Remark
Table	1	DDL for table creation
View	1	Includes Materialized views
Standalone SQL	1	
Bteq	1	In case of bteq called from shell or Python scripts, a multiplication factor of 10 is applied to the object count
SP	2	
Macros	1	
TPT	2	
Fastload	1	
Mload	1	
Functions	1	
UDF	2	

Weightage is applied to the object count in order to get the final raven-object-count for usage and billing.

Raven-object-count = object-count * weightage

Following are some examples:

Eg. 1 . DDL for Tables creation

```
CREATE TABLE emp (          ------- object 1
emp_id DECIMAL(18,0), first_name char(10), Last_name char(10),
);
```

Weightage for DDL for table creation is 1.

Raven-object-count = 1 * 1

In this case the final raven-object-count =1

Eg. 2. Stored procedure

```
CREATE PROCEDURE InsertSalary ( IN in_EmployeeNo INTEGER, IN in_Gross INTEGER)
BEGIN
  INSERT INTO Salary ( EmployeeNo, Gross)  ----- Object 1
  VALUES ( :in_EmployeeNo, :in_Gross )
);
END;
```

Weightage for SP is 2

Raven-object-count = 1 * 2

In this case the final raven-object-count =2

Eg. 3. Standalone SQL statement

```
SELECT * FROM trimmed_salary e  ---------object 1
INNER JOIN
(SELECT * FROM department ---------object 2
) d
ON e.department_id = d.department_id
```

Weightage for DDL for table creation is 1.

Raven-object-count = 2 * 1

In this case the final raven-object-count =2