CPRG 307

Syntax Reference Guide

All quizzes and exams in CPRG 307 are closed book.

This syntax reference guide will be provided to you by your instructor at the start of each quiz and exam to be used as a resource.

Note that this reference guide provides standard structures but does **not** provide **all** coding options with these structures as some variations are what you must demonstrate on the quiz or exam.

Contents

SQL Commands

DDL

CREATE TABLE

```
CREATE TABLE table_name
(column_name    datatype,
 column_name    datatype,
.
.
.
);
```

ALTER TABLE (ADDING COLUMNS)

```
ALTER TABLE table_name
ADD column_name datatype;
```

DROP TABLE

```
DROP TABLE table_name;
```

DML

INSERT

```
INSERT INTO table_name
[(column_name1,column_name2,. . .)]
VALUES
(value1,value2,. . .);
```

UPDATE

```
UPDATE table_name
   SET column_name = value,
       column_name = value
[WHERE condition];
```

DELETE

```
DELETE [FROM] table_name
[WHERE condition];
```

DQL

WITHOUT JOIN

```
SELECT column1,column2,. . .
  FROM table_name
[WHERE condition]
[GROUP BY column1,column2,. . .]
[HAVING condition]
[ORDER BY column1,column2,. . .];
```

WITH EQUI-JOIN (JOIN FORM)

```
SELECT column1,column2,. . .
  FROM table_name1 t1 JOIN table_name2 t2
                        ON t1.column_name = t2.column_name
                      JOIN table_name3 t3
                        ON t2.column_name = t3.column_name
                           .
                           .
                           .
[WHERE condition]
[GROUP BY column1,column2,. . .]
[HAVING condition]
[ORDER BY column1,column2,. . .];
```

WITH EQUI-JOIN (TRADITIONAL METHOD)

```
SELECT column1,column2,. . .
  FROM table_name1 t1, table_name2 t2, table_name3 t3,
                              .
                              .
                              .
 WHERE t1.column_name = t2.column_name
   AND t2.column_name = t3.column_name
                              .
                              .
                              .
[GROUP BY column1,column2,. . .]
[HAVING condition]
[ORDER BY column1,column2,. . .];
```

PL/SQL

Anonymous Block

```
[DECLARE]
  -- variable declaration area

BEGIN
  -- code functionality / logic

[EXCEPTION]
  -- exception conditions

END;
/
```

Stored Programs

PROCEDURE

```
CREATE [OR REPLACE] PROCEDURE procedure_name [(parameter1,parameter2,.
. .)] IS


BEGIN

  -- code functionality


[EXCEPTION]

  -- exception conditions


END;

/
```

FUNCTION

```
CREATE OR REPLACE FUNCTION function_name
[(parameter1, parameter2,. . .)] RETURN datatype IS

BEGIN
  -- code functionality
  RETURN value;

[EXCEPTION]
  -- exception conditions

END;
/
```

DML TRIGGER

```
CREATE [OR REPLACE] TRIGGER trigger_name
{BEFORE | AFTER}
{INSERT | UPDATE | DELETE | multiple}
[OF column_name]
ON table_name
[FOR EACH ROW]
[WHEN condition]

[DECLARE]
  -- variable declaration area
BEGIN
  -- code functionality

[EXCEPTION]
  -- exception conditions
END;
/
```

Code Structures

STANDARD DATATYPES

```
DATE

NUMBER(p,s)

VARCHAR2(n)

CHAR(n)

BOOLEAN -- PL/SQL only
```

STANDARD BUILT-IN FUNCTIONS

```
SYSDATE

TO_DATE('literal_date_value','date format')

TO_CHAR(date_column,'date format')

AVG(column_name)

SUM(column_name)

COUNT(*)

COUNT(column_name)
```

IF

```
IF (condition) THEN
    -- code functionality
ELSIF (condition) THEN
    -- code functionality
.
.
.
[ELSE]
    -- code functionality
END IF;
```

CASE

```
CASE
  WHEN (condition) THEN
      -- code functionality
  WHEN (condition) THEN
      -- code functionality
  .
  .
  .
  [ELSE]
      -- code functionality
  END CASE;
```

BASIC LOOP (NO CURSOR)

```
LOOP
  EXIT WHEN (condition);
  -- functionality that changes the condition over time
  -- code functionality
END LOOP;
```

FOR LOOP (NO CURSOR)

```
FOR variable IN start_value..end_value LOOP
  -- code functionality
END LOOP;
```