

ORACLE SQL Syntaxes

S/N	Name	Syntax
1.	SELECT	<code>SELECT column1, column2, ... FROM table_name;</code>
2.	WHERE	<code>SELECT column1, column2, ... FROM table_name WHERE condition;</code>
3.	ORDER BY	<code>SELECT column1, column2, ... FROM table_name ORDER BY column1, column2, ... ASC DESC;</code>
4.	SELECT DISTINCT	<code>SELECT DISTINCT column1, column2, ... FROM table_name;</code>
5.	AND	<code>SELECT column1, column2, ... FROM table_name WHERE condition1 AND condition2 AND condition3 ...;</code>
6.	OR	<code>SELECT column1, column2, ... FROM table_name WHERE condition1 OR condition2 OR condition3 ...;</code>
7.	LIKE	<code>SELECT column1, column2, ... FROM table_name WHERE columnN LIKE pattern;</code>
8.	IN	<code>SELECT column_name(s) FROM table_name WHERE column_name IN (value1, value2, ...);</code>
9.	BETWEEN	<code>SELECT column_name(s) FROM table_name WHERE column_name BETWEEN value1 AND value2;</code>
10.	COLUMN ALIAS	<code>SELECT column_name AS alias_name FROM table_name;</code> <i>Or</i> <code>SELECT column_name "alias_name" FROM table_name;</code>
11.	Case Manipulation Functions	<code>UPPER(column_name expresssion) LOWER(column_name expresssion) INITCAP(column_name expresssion)</code>

12.	Character Manipulation Functions	<code>CONCAT(column1 expression1, column2 expression2)</code> <code>SUBSTR(column_name expression, starting position, length)</code> <code>LENGTH(column_name expression)</code> <code>LPAD(column_name expression, total number of characters in the padded string, the character to pad with)</code> <code>INSTR(column_name expression, column_name expression)</code>	
13.	Additional Manipulation Functions - TRIM	<code>TRIM([{ { LEADING TRAILING BOTH } [trim_character] trim_character } FROM] trim_source)</code> E.g. <code>SELECT TRIM(LEADING 'a' FROM 'abcba')</code> <code>FROM DUAL;</code> Result: bcba	
14.	Additional Manipulation Functions - REPLACE	<code>REPLACE (string1, string_to_replace, [replacement_string])</code>	
15.	Number Functions	<code>ROUND (column/expression, decimal places)</code> <code>TRUNC(column/expression, decimal places)</code> <code>MOD(column/expression, number)</code>	
16.	DATE FUNCTIONS	<div>MONTHS_BETWEEN</div>	Number of months between two dates <code>MONTHS_BETWEEN(date_value1, date_value2)</code>

		ADD_MONTHS	Add calendar months to date <code>ADD_MONTHS(date_value, number)</code>
		NEXT_DAY	Date of the next occurrence of day of the week specified <code>NEXT_DAY(date_value, weekday)</code>
		LAST_DAY	Last day of the month <code>LAST_DAY(date_value)</code>
		ROUND	Round date <code>ROUND(date_value, unit of date)</code>
		TRUNC	Truncate date <code>TRUNC(date_value, unit of date)</code>
17.	Type Conversion Functions	<code>TO_CHAR(date column name, 'format model you specify')</code> <code>TO_CHAR(number, 'format model')</code> <code>TO_NUMBER(character string, 'format model')</code> <code>TO_DATE(character string, 'format model')</code>	
18.	NULL Functions	<code>NVL(expression 1 value that may contain a null, expression 2 value to substitute for null)</code> <code>NVL2(expression 1 value that may contain a null, expression 2 value to return if expression 1 is not null, expression 3 value to replace if expression 1 is null)</code> <code>NULLIF(expression 1, expression 2)</code> <code>COALESCE(expression 1, expression 2, ...expression n)</code>	
19.	Conditional Expressions - CASE	<code>CASE</code> <code>WHEN condition1 THEN result1</code> <code>WHEN condition2 THEN result2</code> <code>WHEN conditionN THEN resultN</code> <code>ELSE result</code> <code>END;</code>	

20.	Conditional Expressions - DECODE	<code>DECODE(column1 expression, search1, result1 [, search2, result2,...,] [, default])</code>
21.	INNER JOIN	<code>SELECT column_name(s) FROM table1 INNER JOIN table2 ON table1.column_name = table2.column_name;</code>
22.	LEFT OUTER JOIN	<code>SELECT column_name(s) FROM table1 LEFT OUTER JOIN table2 ON table1.column_name = table2.column_name;</code>
23.	RIGHT OUTER JOIN	<code>SELECT column_name(s) FROM table1 RIGHT OUTER JOIN table2 ON table1.column_name = table2.column_name;</code>
24.	GROUP BY	<code>SELECT column_name(s) FROM table_name WHERE condition GROUP BY column_name(s);</code>
25.	HAVING	<code>SELECT column_name(s) FROM table_name WHERE condition GROUP BY column_name(s) HAVING condition ORDER BY column_name(s);</code>
26.		<code>SELECT column_name(s) FROM table_name WHERE column_name operator (SELECT column_name(s) FROM table_name [WHERE] condition)</code>

The diagram illustrates the database schema for the 'SCOTT' schema, showing the relationships between various tables and their attributes.

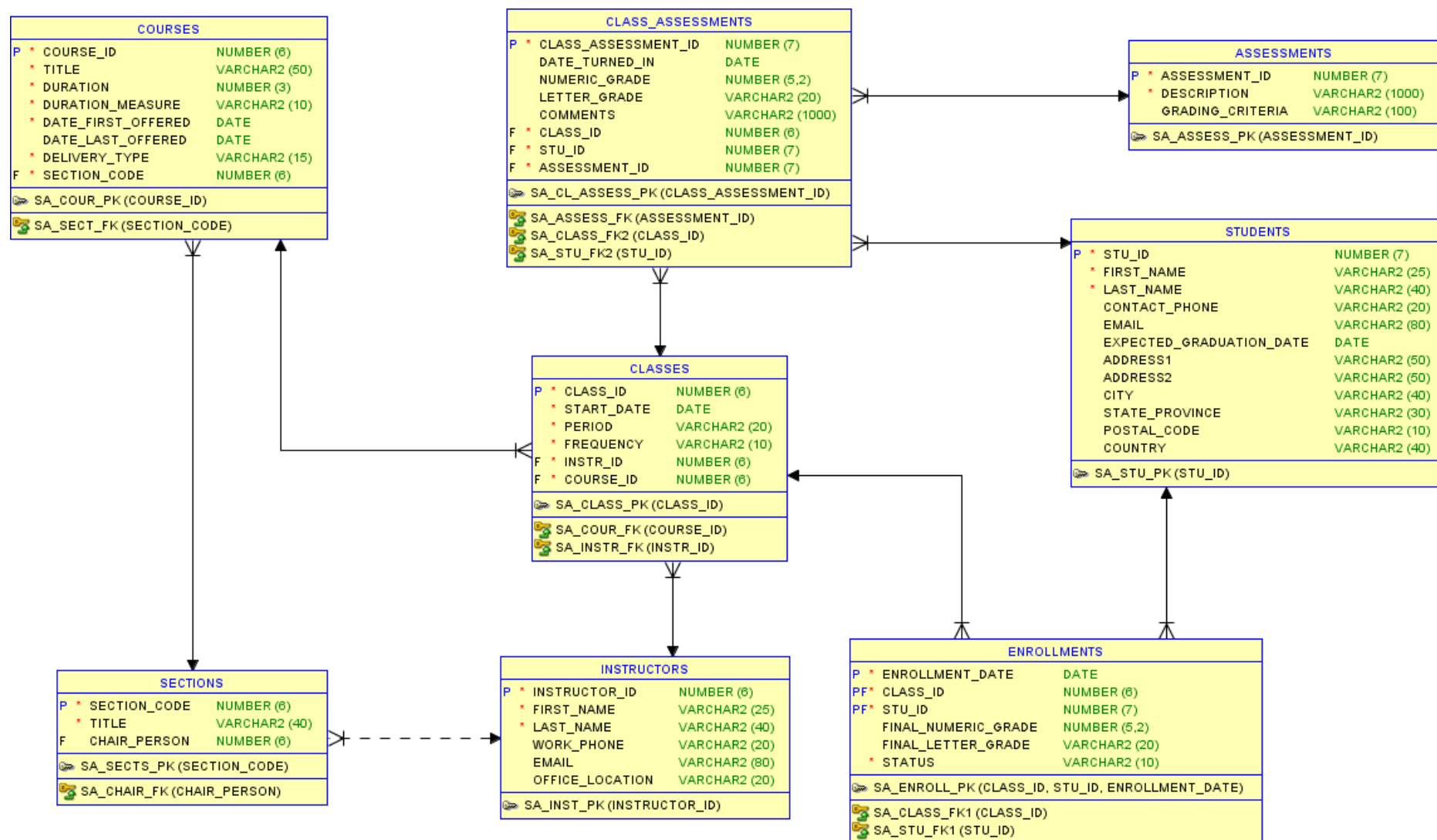
Tables and Attributes:

- EMPLOYEES:** EMPLOYEE_ID (P, NUMBER(6)), FIRST_NAME (VARCHAR2(20)), LAST_NAME (F, VARCHAR2(25)), EMAIL (U, VARCHAR2(25)), PHONE_NUMBER (VARCHAR2(20)), HIRE_DATE (DATE), JOB_ID (F, VARCHAR2(10)), SALARY (NUMBER(8,2)), COMMISSION_PCT (NUMBER(2,2)), MANAGER_ID (F, NUMBER(6)), DEPARTMENT_ID (F, NUMBER(4)), BONUS (VARCHAR2(5)). Constraints: EMP_ID_PK (EMPLOYEE_ID), EMP_EMAIL_UK (EMAIL), EMP_DEPT_FK (DEPARTMENT_ID), EMP_JOB_FK (JOB_ID), EMP_MANAGER_FK (MANAGER_ID), EMP_DEPARTMENT_IX (DEPARTMENT_ID), EMP_JOB_IX (JOB_ID), EMP_MANAGER_IX (MANAGER_ID), EMP_NAME_IX (LAST_NAME, FIRST_NAME).
- COUNTRIES:** COUNTRY_ID (P, NUMBER(4)), REGION_ID (F, NUMBER(3)), COUNTRY_NAME (F, VARCHAR2(70)), COUNTRY_TRANSLATED_NAME (VARCHAR2(40)), LOCATION (VARCHAR2(80)), CAPITOL (VARCHAR2(50)), AREA (NUMBER(15)), COASTLINE (NUMBER(8)), LOWEST_ELEVATION (NUMBER(6)), LOWEST_ELEV_NAME (VARCHAR2(70)), HIGHEST_ELEVATION (NUMBER(6)), HIGHEST_ELEV_NAME (VARCHAR2(50)), DATE_OF_INDEPENDENCE (VARCHAR2(30)), NATIONAL_HOLIDAY_NAME (VARCHAR2(200)), NATIONAL_HOLIDAY_DATE (VARCHAR2(30)), POPULATION (NUMBER(12)), POPULATION_GROWTH_RATE (VARCHAR2(10)), LIFE_EXPECT_AT_BIRTH (NUMBER(6,2)), MEDIAN_AGE (NUMBER(6,2)), AIRPORTS (NUMBER(6)), CLIMATE (VARCHAR2(1000)), FIPS_ID (CHAR(2)), INTERNET_EXTENSION (VARCHAR2(3)), FLAG (BLOB), CURRENCY_CODE (F, VARCHAR2(7)). Constraints: CTRY_PK (COUNTRY_ID), COUNTRY_REG_ID_FK1 (REGION_ID), CTRY_CURR_FK (CURRENCY_CODE), CTRY_REG_IDX (REGION_ID).
- CURRENCIES:** CURRENCY_CODE (P, VARCHAR2(7)), CURRENCY_NAME (F, VARCHAR2(40)), COMMENTS (VARCHAR2(150)). Constraint: CURR_PK (CURRENCY_CODE).
- REGIONS:** REGION_ID (P, NUMBER(3)), REGION_NAME (F, VARCHAR2(35)). Constraint: REG_ID_PK (REGION_ID).
- SPOKEN_LANGUAGES:** COUNTRY_ID (PF, NUMBER(4)), LANGUAGE_ID (PF, NUMBER(4)), OFFICIAL (F, VARCHAR2(2)), COMMENTS (F, VARCHAR2(512)). Constraints: SPOKEN_LANG_PK (COUNTRY_ID, LANGUAGE_ID), CTRY_NUM_FK1 (COUNTRY_ID), LANG_ID_FK2 (LANGUAGE_ID).
- LANGUAGES:** LANGUAGE_ID (P, NUMBER(4)), LANGUAGE_NAME (F, VARCHAR2(50)). Constraint: LANG_PK (LANGUAGE_ID).
- JOBS:** JOB_ID (P, VARCHAR2(10)), JOB_TITLE (F, VARCHAR2(35)), MIN_SALARY (NUMBER(6)), MAX_SALARY (NUMBER(6)). Constraint: JOB_ID_PK (JOB_ID).
- JOB_GRADES:** GRADE_LEVEL (F, VARCHAR2(3)), LOWEST_SAL (NUMBER), HIGHEST_SAL (NUMBER).
- JOB_HISTORY:** EMPLOYEE_ID (P, NUMBER(6)), START_DATE (P, DATE), END_DATE (F, DATE), JOB_ID (F, VARCHAR2(10)), DEPARTMENT_ID (F, NUMBER(4)). Constraints: JHIST_EMP_ID_ST_DATE_PK (EMPLOYEE_ID, START_DATE), JHIST_DEPT_FK (DEPARTMENT_ID), JHIST_JOB_FK (JOB_ID), JHIST_DEPARTMENT_IX (DEPARTMENT_ID), JHIST_EMPLOYEE_IX (EMPLOYEE_ID), JHIST_JOB_IX (JOB_ID).
- DEPARTMENTS:** DEPARTMENT_ID (P, NUMBER(4)), DEPARTMENT_NAME (F, VARCHAR2(30)), MANAGER_ID (F, NUMBER(6)), LOCATION_ID (F, NUMBER(4)). Constraints: DEPT_ID_PK (DEPARTMENT_ID), DEPT_LOC_FK (LOCATION_ID), DEPT_MGR_FK (MANAGER_ID), DEPT_LOCATION_IX (LOCATION_ID).
- LOCATIONS:** LOCATION_ID (P, NUMBER(4)), STREET_ADDRESS (F, VARCHAR2(40)), POSTAL_CODE (F, VARCHAR2(12)), CITY (F, VARCHAR2(30)), STATE_PROVINCE (F, VARCHAR2(25)), COUNTRY_ID (F, NUMBER(4)). Constraints: LOC_ID_PK (LOCATION_ID), LOC_C_ID_FK (COUNTRY_ID), LOC_CITY_IX (CITY), LOC_COUNTRY_IX (COUNTRY_ID), LOC_STATE_PROVINCE_IX (STATE_PROVINCE).

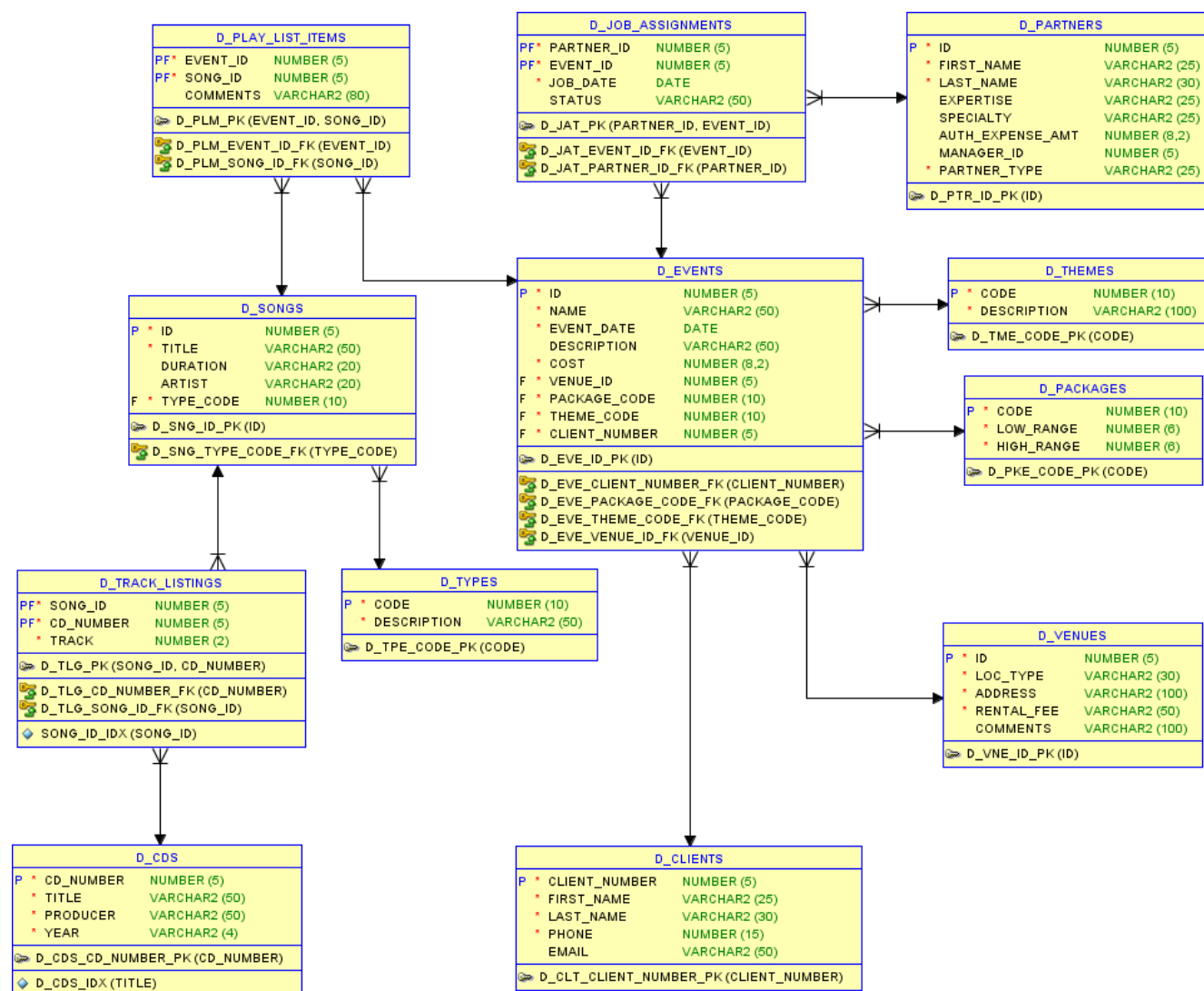
Relationships:

- EMPLOYEES to COUNTRIES: One-to-many (EMPLOYEE_ID to COUNTRY_ID).
- COUNTRIES to CURRENCIES: One-to-many (COUNTRY_ID to CURRENCY_CODE).
- COUNTRIES to REGIONS: One-to-many (COUNTRY_ID to REGION_ID).
- COUNTRIES to SPOKEN_LANGUAGES: One-to-many (COUNTRY_ID to COUNTRY_ID).
- SPOKEN_LANGUAGES to LANGUAGES: One-to-many (LANGUAGE_ID to LANGUAGE_ID).
- JOBS to EMPLOYEES: One-to-many (JOB_ID to JOB_ID).
- JOB_HISTORY to EMPLOYEES: One-to-many (EMPLOYEE_ID to EMPLOYEE_ID).
- JOB_HISTORY to DEPARTMENTS: One-to-many (DEPARTMENT_ID to DEPARTMENT_ID).
- JOB_HISTORY to JOBS: One-to-many (JOB_ID to JOB_ID).
- DEPARTMENTS to EMPLOYEES: One-to-many (DEPARTMENT_ID to DEPARTMENT_ID).
- DEPARTMENTS to LOCATIONS: One-to-many (LOCATION_ID to LOCATION_ID).
- LOCATIONS to EMPLOYEES: One-to-many (LOCATION_ID to LOCATION_ID).

Database Schema for School Database



Database Schema for DJs on Demand



Database Schema for Global Fast Foods

