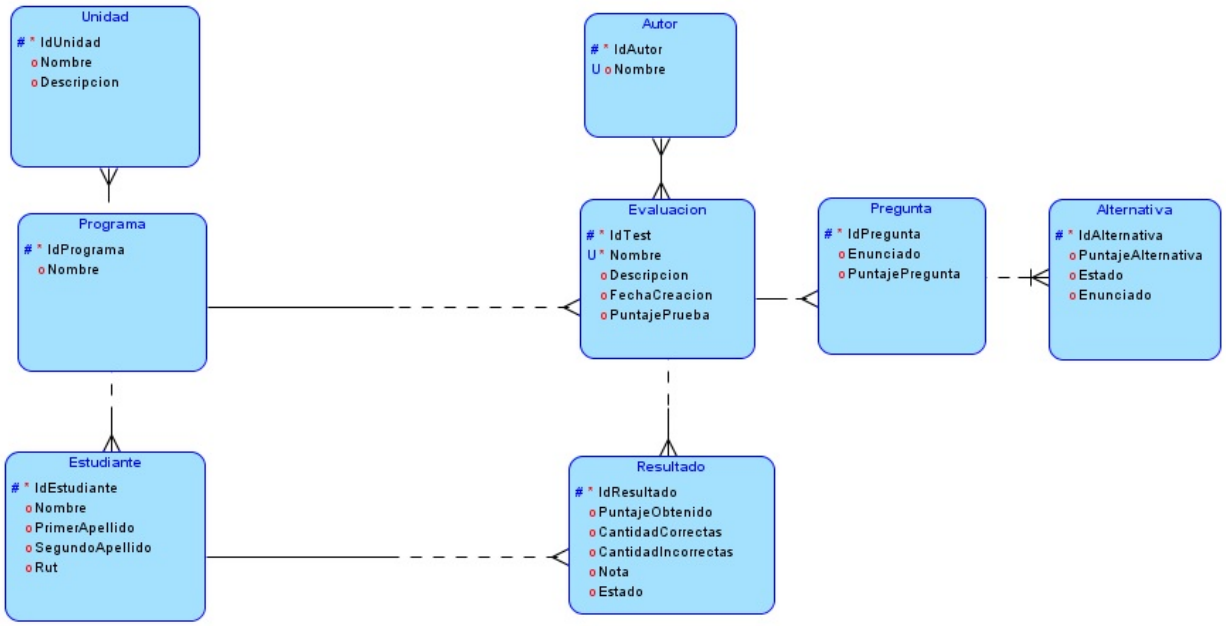
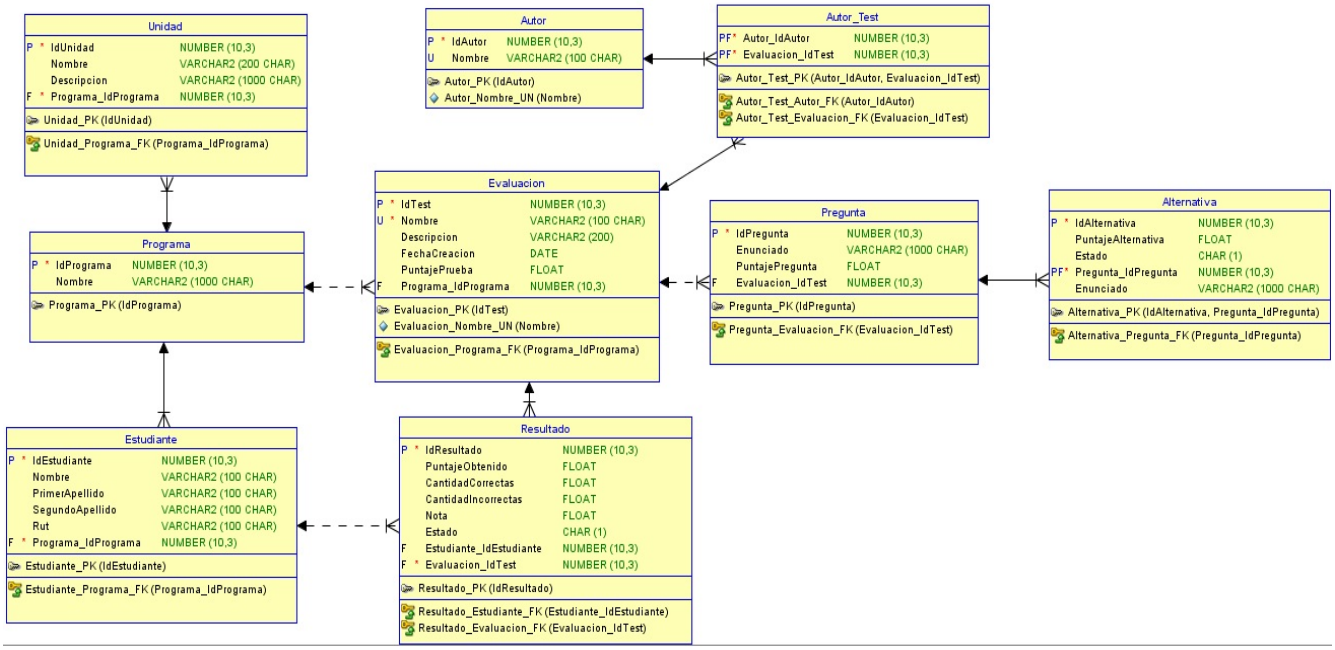
Desarrollo Evaluación cierre módulo 2

Nombre: Katherine Ramos Larenas
Curso: FullStack Java Rancagua

1- Creación modelo lógico en OracleDataModeler



2- Modelo Relacional generado por OracleDataModeler



3-DDL generado por OracleDataModeler

```
CREATE TABLE alternativa (
    idalternativa        NUMBER(10, 3) NOT NULL,
    puntajealternativa   FLOAT,
    estado               CHAR(1),
    pregunta_idpregunta  NUMBER(10, 3) NOT NULL,
    enunciado            VARCHAR2(1000 CHAR)
);

ALTER TABLE alternativa ADD CONSTRAINT alternativa_pk PRIMARY KEY ( idalternativa,
                                                                    pregunta_idpregunta );

CREATE TABLE autor (
    idautor  NUMBER(10, 3) NOT NULL,
    nombre   VARCHAR2(100 CHAR)
);

ALTER TABLE autor ADD CONSTRAINT autor_pk PRIMARY KEY ( idautor );

ALTER TABLE autor ADD CONSTRAINT autor_nombre_un UNIQUE ( nombre );

CREATE TABLE autor_test (
    autor_idautor      NUMBER(10, 3) NOT NULL,
    evaluacion_idtest  NUMBER(10, 3) NOT NULL
);

ALTER TABLE autor_test ADD CONSTRAINT autor_test_pk PRIMARY KEY ( autor_idautor,
                                                                  evaluacion_idtest );

CREATE TABLE estudiante (
    idestudiante         NUMBER(10, 3) NOT NULL,
    nombre               VARCHAR2(100 CHAR),
    primerapellido       VARCHAR2(100 CHAR),
    segundoapellido      VARCHAR2(100 CHAR),
    rut                  VARCHAR2(100 CHAR),
    programa_idprograma  NUMBER(10, 3) NOT NULL
);

ALTER TABLE estudiante ADD CONSTRAINT estudiante_pk PRIMARY KEY ( idestudiante );

CREATE TABLE evaluacion (
    idtest               NUMBER(10, 3) NOT NULL,
    nombre               VARCHAR2(100 CHAR) NOT NULL,
    descripcion          VARCHAR2(200),
    fechacreacion        DATE,
    puntajeprueba        FLOAT,
    programa_idprograma  NUMBER(10, 3)
);

ALTER TABLE evaluacion ADD CONSTRAINT evaluacion_pk PRIMARY KEY ( idtest );

ALTER TABLE evaluacion ADD CONSTRAINT evaluacion_nombre_un UNIQUE ( nombre );

CREATE TABLE pregunta (
    idpregunta         NUMBER(10, 3) NOT NULL,
    enunciado          VARCHAR2(1000 CHAR),
```

```
    puntajepregunta    FLOAT,
    evaluacion_idtest  NUMBER(10, 3)
);

ALTER TABLE pregunta ADD CONSTRAINT pregunta_pk PRIMARY KEY ( idpregunta );

CREATE TABLE programa (
    idprograma  NUMBER(10, 3) NOT NULL,
    nombre      VARCHAR2(1000 CHAR)
);

ALTER TABLE programa ADD CONSTRAINT programa_pk PRIMARY KEY ( idprograma );

CREATE TABLE resultado (
    idresultado             NUMBER(10, 3) NOT NULL,
    puntajeobtenido         FLOAT,
    cantidadcorrectas       FLOAT,
    cantidadincorrectas     FLOAT,
    nota                    FLOAT,
    estado                  CHAR(1),
    estudiante_idestudiante  NUMBER(10, 3),
    evaluacion_idtest       NUMBER(10, 3) NOT NULL
);

ALTER TABLE resultado ADD CONSTRAINT resultado_pk PRIMARY KEY ( idresultado );

CREATE TABLE unidad (
    idunidad            NUMBER(10, 3) NOT NULL,
    nombre              VARCHAR2(200 CHAR),
    descripcion         VARCHAR2(1000 CHAR),
    programa_idprograma  NUMBER(10, 3) NOT NULL
);

ALTER TABLE unidad ADD CONSTRAINT unidad_pk PRIMARY KEY ( idunidad );

ALTER TABLE alternativa
    ADD CONSTRAINT alternativa_pregunta_fk FOREIGN KEY ( pregunta_idpregunta )
        REFERENCES pregunta ( idpregunta );

ALTER TABLE autor_test
    ADD CONSTRAINT autor_test_autor_fk FOREIGN KEY ( autor_idautor )
        REFERENCES autor ( idautor );

ALTER TABLE autor_test
    ADD CONSTRAINT autor_test_evaluacion_fk FOREIGN KEY ( evaluacion_idtest )
        REFERENCES evaluacion ( idtest );

ALTER TABLE estudiante
    ADD CONSTRAINT estudiante_programa_fk FOREIGN KEY ( programa_idprograma )
        REFERENCES programa ( idprograma );

ALTER TABLE evaluacion
    ADD CONSTRAINT evaluacion_programa_fk FOREIGN KEY ( programa_idprograma )
        REFERENCES programa ( idprograma );

ALTER TABLE pregunta
    ADD CONSTRAINT pregunta_evaluacion_fk FOREIGN KEY ( evaluacion_idtest )
        REFERENCES evaluacion ( idtest );
```

```
ALTER TABLE resultado
   ADD CONSTRAINT resultado_estudiante_fk FOREIGN KEY ( estudiante_idestudiante )
      REFERENCES estudiante ( idestudiante );

ALTER TABLE resultado
   ADD CONSTRAINT resultado_evaluacion_fk FOREIGN KEY ( evaluacion_idtest )
      REFERENCES evaluacion ( idtest );

ALTER TABLE unidad
   ADD CONSTRAINT unidad_programa_fk FOREIGN KEY ( programa_idprograma )
      REFERENCES programa ( idprograma );
```

4-Script SQL

```
--Asignar valores a programas(cursos)
INSERT ALL
INTO programa VALUES(1,'Curso Java')
INTO programa VALUES(2,'Curso Android')
INTO programa VALUES(3,'Curso Python')
SELECT * FROM dual;

--Asignar valores a unidades y asociarlas a un programa mediante foreign key(FK)
INSERT ALL
INTO unidad VALUES(1, 'Introducción POO', 'Se definen conceptos generales de Programación orientada a objeto',1)
INTO unidad VALUES(2, 'Historia Android', 'Se discute sobre la creación de android y posicionamiento en mercado actual',2)
SELECT * FROM dual;

--Asignar valores a estudiantes
INSERT ALL
INTO estudiante VALUES(1,'Manuel','Flores','Farfán','18.104.524-8',1)
INTO estudiante VALUES(2,'Matías','Castañeda','','19.234.743-k',1)
INTO estudiante VALUES(3,'Miguel','Rubilar','Bravo','23.789.432-2',1)
INTO estudiante VALUES(4,'Camila','Fuentes','Fuentes','12.546.293-9',1)
INTO estudiante VALUES(5,'Valentina','Sanchez','Cavieres','20.934.124-6',1)
INTO estudiante VALUES(6,'Félix','Guzmán','López','14.444.198-4',1)
INTO estudiante VALUES(7,'Marta','Carrasco','Valdivia','17.348.534-6',1)
INTO estudiante VALUES(8,'Fernanda','Castro','López','19.294.735-2',1)
INTO estudiante VALUES(9,'Carla','Díaz','Díaz','21.789.234-5',1)
INTO estudiante VALUES(10,'Paloma','Jimenez','','26.231.876-1',1)
INTO estudiante VALUES(11,'Alberto','Rojas','','21.783.527-3',2)
INTO estudiante VALUES(12,'Kevin','Galaz','Ríos','20.927.483-4',2)
INTO estudiante VALUES(13,'Juan','Campos','Campos','18.238.527-2',2)
INTO estudiante VALUES(14,'María','Lastra','Villalón','19.335.879-5',2)
INTO estudiante VALUES(15,'Viviana','Díaz','Cáceres','19.783.996-6',2)
INTO estudiante VALUES(16,'Sara','Pinto','Salas','26.431.876-3',2)
INTO estudiante VALUES(17,'Tania','Zamorano','','12.914.114-2',2)
INTO estudiante VALUES(18,'Lucia','Poblete','Lara','15.238.433-3',2)
INTO estudiante VALUES(19,'Carolina','Poblete','Santis','14.425.658-8',2)
INTO estudiante VALUES(20,'Pedro','Jara','Fernández','18.545.237-7',2)
SELECT * FROM dual;

--Asignar valores a cada evaluacion
INSERT ALL
INTO evaluacion VALUES(1,'Java Básico','Preguntas Java',to_date('14-03-20','dd:mm:yyyy'),100,1)
INTO evaluacion VALUES(2,'SQL Básico','Preguntas SQL',to_date('14-03-20','dd:mm:yyyy'),100,2)
```

```
INTO evaluacion VALUES(3,'Eval D','Preguntas raras',to_date('14-03-20','dd:mm:yyyy'),80,Null)
INTO evaluacion VALUES(4,'Eval V','Ninguna pregunta',to_date('14-03-20','dd:mm:yyyy'),Null,Null)
SELECT * FROM dual;

--Crear un solo autor para todas las evaluaciones
INSERT INTO autor VALUES(1, 'Katherine Ramos Larenas');

--Asociar dicho autor a cada evaluacion mediante sus respectivas FK
INSERT ALL
INTO autor_test VALUES(1,1)
INTO autor_test VALUES(1,2)
INTO autor_test VALUES(1,3)
INTO autor_test VALUES(1,4)
SELECT * FROM dual;

--Asignar valores a preguntas
--Prueba 1: Java Básico
INSERT ALL
INTO pregunta
VALUES(1,'¿Cuál es la descripción que crees que define mejor el concepto clase en la programación orientada
a objetos?',10,1)
INTO pregunta
VALUES(2,'¿Qué elementos crees que definen a un objeto?',10,1)
INTO pregunta
VALUES(3,'¿Qué código de los siguientes tiene que ver con la herencia?',10,1)
INTO pregunta
VALUES(4,'¿Qué significa instanciar una clase?',10,1)
INTO pregunta
VALUES(5,'En Java, ¿a qué nos estamos refiriendo si hablamos de Swing?',10,1)
INTO pregunta
VALUES(6,'¿Qué es Eclipse?',10,1)
INTO pregunta
VALUES(7,'¿Qué es el bytecode en Java?',10,1)
INTO pregunta
VALUES(8,'¿Qué código asociarías a una Interfaz en Java?',10,1)
INTO pregunta
VALUES(9,'¿Qué significa sobrecargar (overload) un método?',10,1)
INTO pregunta
VALUES(10,'¿Qué es una excepción?',10,1)
SELECT * FROM dual;

--Prueba 2: SQL Básico
INSERT ALL
INTO pregunta
VALUES(11,'¿Cómo se obtienen todos los nombres de personas que comienzan con "Juan"?',10,2)
INTO pregunta
VALUES(12,'¿En SQL, para eliminar las filas duplicadas del resultado de una sentencia SELECT se
emplea?',10,2)
INTO pregunta
VALUES(13,'¿Qué instrucción se emplea para eliminar todo el contenido de una tabla, pero conservando la
tabla?',10,2)
INTO pregunta
VALUES(14,'¿Cuál de las siguientes NO es una función de agregación?',10,2)
INTO pregunta
VALUES(15,'En SQL, para ordenar los datos devueltos por una sentencia SELECT se emplea la cláusula:',10,2)
INTO pregunta
VALUES(16,'En SQL, ¿Cuál de estas sentencias añade una fila a una tabla en una base de datos?',10,2)
INTO pregunta
VALUES(17,'¿En cuál de las siguientes sentencias del lenguaje SQL se emplea la cláusula SET?',10,2)
```

```
INTO pregunta
VALUES(18,'En SQL, para modificar la estructura de una tabla de una base de datos se emplea la instrucción:',10,2)
INTO pregunta
VALUES(19,'En una cláusula LIKE, ¿Cómo se obtienen todos los nombres de personas que tienen exactamente cuatro caracteres?',10,2)
INTO pregunta
VALUES(20,'Una sentencia SELECT sin la cláusula WHERE devuelve:',10,2)
SELECT * FROM dual;

--Prueba 3: Eval D(evaluación con deficiencias para ser usada en consultas)
INSERT ALL
INTO pregunta
VALUES(21,'JAVA es un lenguaje de programación:',10,3)
INTO pregunta
VALUES(22,'¿Cuál fue uno de los objetivos principales cuando fue diseñado?',Null,3)
INTO pregunta
VALUES(23,'¿Qué significan las iniciales JDK?',20,3)
INTO pregunta
VALUES(24,'¿Qué es una interface en Java?',Null,3)
INTO pregunta
VALUES(25,'¿Cuál es la diferencia entre una interfaz y una clase abstracta?',Null,3)
INTO pregunta
VALUES(26,'¿Cuál es la estructura para hacer un comentario?',15,3)
INTO pregunta
VALUES(27,'¿Cuáles de las siguientes son palabras reservadas de Java?',20,3)
INTO pregunta
VALUES(28,'¿Cuáles de los siguientes son operadores aritméticos en Java?',15,3)
INTO pregunta
VALUES(29,'A lo menos cuántos métodos tiene una clase',Null,3)
INTO pregunta
VALUES(30,'¿Qué es un objeto en Java?',Null,3)
SELECT * FROM dual;

--Asignar valores a alternativas
--Prueba 1 pregunta 1
INSERT ALL
INTO alternativa
VALUES(1,Null,'F',1,'Es un tipo particular de variable')
INTO alternativa
VALUES(2,10,'V',1,'Es un modelo o plantilla a partir de la cual creamos objetos')
INTO alternativa
VALUES(3,Null,'F',1,'Es una categoria de datos ordenada secuencialmente')
INTO alternativa
VALUES(4,Null,'F',1,'Es un concepto similar al de array')
SELECT * FROM dual;

--Prueba 1 pregunta 2
INSERT ALL
INTO alternativa
VALUES(5,Null,'F',2,'Sus cardinalidad y su tipo')
INTO alternativa
VALUES(6,10,'V',2,'Sus atributos y sus métodos')
INTO alternativa
VALUES(7,Null,'F',2,'Su interfaz y los eventos asociados')
INTO alternativa
VALUES(8,Null,'F',2,'La forma en que establece comunicación e intercambia mensajes')
SELECT * FROM dual;
```

```
--Prueba 1 pregunta 3
INSERT ALL
INTO alternativa
VALUES(9,10,'V',3,'public class Componente extends Producto')
INTO alternativa
VALUES(10,Null,'F',3,'public class Componente inherit Producto')
INTO alternativa
VALUES(11,Null,'F',3,'public class Componente implements Producto')
INTO alternativa
VALUES(12,Null,'F',3,'public class Componente belongs Producto')
SELECT * FROM dual;

--Prueba 1 pregunta 4
INSERT ALL
INTO alternativa
VALUES(13,Null,'F',4,'Duplicar una clase')
INTO alternativa
VALUES(14,Null,'F',4,'Eliminar una clase')
INTO alternativa
VALUES(15,10,'V',4,'Crear un objeto a partir de la clase')
INTO alternativa
VALUES(16,Null,'F',4,'Conectar dos clases entre sí')
SELECT * FROM dual;

--Prueba 1 pregunta 5
INSERT ALL
INTO alternativa
VALUES(17,Null,'F',5,'Una función utilizada para intercambiar valores')
INTO alternativa
VALUES(18,Null,'F',5,'Un framework específico para Android')
INTO alternativa
VALUES(19,Null,'F',5,'Es el sobrenombre de la versión 1.3 del JDK')
INTO alternativa
VALUES(20,10,'V',5,'Una librería para construir interfaces gráficas')
SELECT * FROM dual;

--Prueba 1 pregunta 6
INSERT ALL
INTO alternativa
VALUES(21,Null,'F',6,'Una libreria de Java')
INTO alternativa
VALUES(22,Null,'F',6,'Una versión de Java especial para servidores')
INTO alternativa
VALUES(23,10,'V',6,'Un IDE para desarrollar aplicaciones')
INTO alternativa
VALUES(24,Null,'F',6,'Ninguna de las anteriores')
SELECT * FROM dual;

--Prueba 1 pregunta 7
INSERT ALL
INTO alternativa
VALUES(25,Null,'F',7,'El formato de intercambio de datos')
INTO alternativa
VALUES(26,10,'V',7,'El formato que obtenemos tras compilar un fuente .java')
INTO alternativa
VALUES(27,Null,'F',7,'Un tipo de variable')
INTO alternativa
VALUES(28,Null,'F',7,'Un depurador de código')
SELECT * FROM dual;
```

```
--Prueba 1 pregunta 8
INSERT ALL
INTO alternativa
VALUES(29,5,'V',8,'interface Product')
INTO alternativa
VALUES(30,Null,'F',8,'Componente cp = new Componente (interfaz)')
INTO alternativa
VALUES(31,Null,'F',8,'public class componente interface Product')
INTO alternativa
VALUES(32,5,'V',8,'public class Componente implements Printable')
SELECT * FROM dual;

--Prueba 1 pregunta 9
INSERT ALL
INTO alternativa
VALUES(33,Null,'F',9,'Editarlo para modificar su comportamiento')
INTO alternativa
VALUES(34,Null,'F',9,'Cambiarle el nombre dejandolo con la misma funcionalidad')
INTO alternativa
VALUES(35,10,'V',9,'Crear un método con el mismo nombre pero diferentes argumentos')
INTO alternativa
VALUES(36,Null,'F',9,'Añadirle funcionalidades a un método')
SELECT * FROM dual;

--Prueba 1 pregunta 10
INSERT ALL
INTO alternativa
VALUES(37,5,'V',10,'Un error que lanza un método cuando algo va mal')
INTO alternativa
VALUES(38,Null,'F',10,'Un objeto que no puede ser instanciado')
INTO alternativa
VALUES(39,Null,'F',10,'Un bucle que no finaliza')
INTO alternativa
VALUES(40,5,'V',10,'Un error en tiempo de ejecución-runtime')
SELECT * FROM dual;

--Prueba 2 pregunta 1
INSERT ALL
INTO alternativa
VALUES(41,Null,'F',11,'LIKE "Juan$"')
INTO alternativa
VALUES(42,Null,'F',11,'LIKE "Juan*"')
INTO alternativa
VALUES(43,10,'V',11,'LIKE "Juan%"')
INTO alternativa
VALUES(44,Null,'F',11,'"Juan&"')
SELECT * FROM dual;

--Prueba 2 pregunta 2
INSERT ALL
INTO alternativa
VALUES(45,Null,'F',12,'NO DUPLICATE')
INTO alternativa
VALUES(46,10,'V',12,'DISTINCT')
INTO alternativa
VALUES(47,Null,'F',12,'UNIQUE')
INTO alternativa
VALUES(48,Null,'F',12,'Ninguna de las anteriores')
```

```
SELECT * FROM dual;

--Prueba 2 pregunta 3
INSERT ALL
INTO alternativa
VALUES(49,Null,'F',13,'DELETE TABLE')
INTO alternativa
VALUES(50,10,'V',13,'TRUNCATE TABLE')
INTO alternativa
VALUES(51,Null,'F',13,'DROP TABLE')
INTO alternativa
VALUES(52,Null,'F',13,'ERASE TABLE')
SELECT * FROM dual;

--Prueba 2 pregunta 4
INSERT ALL
INTO alternativa
VALUES(53,Null,'F',14,'COUNT()')
INTO alternativa
VALUES(54,10,'V',14,'LIMIT()')
INTO alternativa
VALUES(55,Null,'F',14,'MAX()')
INTO alternativa
VALUES(56,Null,'F',14,'MIN()')
SELECT * FROM dual;

--Prueba 2 pregunta 5
INSERT ALL
INTO alternativa
VALUES(57,10,'V',15,'ORDER BY')
INTO alternativa
VALUES(58,Null,'F',15,'ORDERER BY')
INTO alternativa
VALUES(59,Null,'F',15,'SORT BY')
INTO alternativa
VALUES(60,Null,'F',15,'SORTED BY')
SELECT * FROM dual;

--Prueba 2 pregunta 6
INSERT ALL
INTO alternativa
VALUES(61,Null,'F',16,'ADD')
INTO alternativa
VALUES(62,10,'V',16,'INSERT')
INTO alternativa
VALUES(63,Null,'F',16,'UPDATE')
INTO alternativa
VALUES(64,Null,'F',16,'INCLUDE')
SELECT * FROM dual;

--Prueba 2 pregunta 7
INSERT ALL
INTO alternativa
VALUES(65,Null,'F',17,'DELETE')
INTO alternativa
VALUES(66,Null,'F',17,'DROP')
INTO alternativa
VALUES(67,Null,'F',17,'SELECT')
INTO alternativa
```

```
VALUES(68,10,'V',17,'UPDATE')
SELECT * FROM dual;

--Prueba 2 pregunta 8
INSERT ALL
INTO alternativa
VALUES(69,10,'V',18,'ALTER TABLE')
INTO alternativa
VALUES(70,Null,'F',18,'CHANGE TABLE')
INTO alternativa
VALUES(71,Null,'F',18,'MODIFY TABLE')
INTO alternativa
VALUES(72,Null,'F',18,'Ninguna de las anteriores')
SELECT * FROM dual;

--Prueba 2 pregunta 9
INSERT ALL
INTO alternativa
VALUES(73,Null,'F',19,'LIKE “$$$$”')
INTO alternativa
VALUES(74,10,'V',19,'LIKE “___”')
INTO alternativa
VALUES(75,Null,'F',19,'LIKE “....”')
INTO alternativa
VALUES(76,Null,'F',19,'LIKE ”////”')
SELECT * FROM dual;

--Prueba 2 pregunta 10
INSERT ALL
INTO alternativa
VALUES(77,5,'V',20,'Una columna especificada después del select')
INTO alternativa
VALUES(78,Null,'F',20,'No se puede utilizar SELECT sin clausula WHERE')
INTO alternativa
VALUES(79,Null,'F',20,'Un bucle que no finaliza')
INTO alternativa
VALUES(80,5,'V',20,'Todos los registros existentes en la tabla')
SELECT * FROM dual;

--Prueba 3 pregunta 1
INSERT ALL
INTO alternativa
VALUES(81,10,'V',21,'Orientado a objetos')
INTO alternativa
VALUES(82,Null,'F',21,'No orientado')
SELECT * FROM dual;

--Prueba 3 pregunta 2
INSERT ALL
INTO alternativa
VALUES(83,Null,'F',22,'Que fuera sencillo para aprender')
INTO alternativa
VALUES(84,Null,'F',22,'Que tuviera más utilidades de programación a bajo nivel que C o C++')
INTO alternativa
VALUES(85,Null,'F',22, 'Que fuera orientado a la programación de servicios web')
SELECT * FROM dual;

--Prueba 3 pregunta 3
INSERT ALL
```

```
INTO alternativa
VALUES(86,5,'V',23,'Java Development Kit')
INTO alternativa
VALUES(87,5,'V',23,'Java Development Kit')
INTO alternativa
VALUES(88,5,'V',23,'Java Development Kit')
INTO alternativa
VALUES(89,5,'V',23,'Java Development Kit')
SELECT * FROM dual;

--Prueba 3 pregunta 4
INSERT ALL
INTO alternativa
VALUES(90,Null,'F',24,'Es un sinónimo de clase')
INTO alternativa
VALUES(91,Null,'F',24,'Es un objeto')
INTO alternativa
VALUES(92,Null,'F',24, 'Todas las anteriores')
SELECT * FROM dual;

--Prueba 3 pregunta 5
INSERT ALL
INTO alternativa
VALUES(93,Null,'F',25,'Las clases abstractas pueden contener variables o métodos privados')
INTO alternativa
VALUES(94,Null,'F',25,'Las interfaces no pueden ser instanciadas pero las clases abstractas sí')
SELECT * FROM dual;


--Prueba 3 pregunta 6
INSERT ALL
INTO alternativa
VALUES(95,5,'V',26,'//')
INTO alternativa
VALUES(96,5,'V',26,'/* */')
INTO alternativa
VALUES(97,5,'V',26,'/** */')
SELECT * FROM dual;

--Prueba 3 pregunta 7
INSERT ALL
INTO alternativa
VALUES(98,5,'V',27,'Null')
INTO alternativa
VALUES(99,5,'V',27,'Static')
INTO alternativa
VALUES(100,5,'V',27,'Public')
INTO alternativa
VALUES(101,5,'V',27,'Return')
SELECT * FROM dual;

--Prueba 3 pregunta 8
INSERT ALL
INTO alternativa
VALUES(102,5,'V',28,'+')
INTO alternativa
VALUES(103,5,'V',28,'-')
INTO alternativa
VALUES(104,5,'V',28,'*')
```

```
SELECT * FROM dual;

--Prueba 3 pregunta 9
INSERT INTO alternativa
VALUES(105,Null,'F',29,'Cinco');

--Prueba 3 pregunta 10
INSERT ALL
INTO alternativa
VALUES(106,Null,'F',30,'Una agrupación de clases')
INTO alternativa
VALUES(107,Null,'F',30,'Una interface')
SELECT * FROM dual;

--Crear resultados
--Curso Java, evaluacion Java
INSERT ALL
INTO resultado VALUES(1,90,9,1,6.3,'A',1,1)
INTO resultado VALUES(2,80,8,2,5.6,'A',2,1)
INTO resultado VALUES(3,60,6,4,4.2,'A',3,1)
INTO resultado VALUES(4,50,5,5,3.5,'R',4,1)
INTO resultado VALUES(5,30,3,7,2.1,'R',5,1)
INTO resultado VALUES(6,90,9,1,6.3,'A',6,1)
INTO resultado VALUES(7,80,8,2,5.6,'A',7,1)
INTO resultado VALUES(8,60,6,4,4.2,'A',8,1)
INTO resultado VALUES(9,50,5,5,3.5,'R',9,1)
INTO resultado VALUES(10,30,3,7,2.1,'R',10,1)
SELECT * FROM dual;

--Curso Android, evaluacion SQL
INSERT ALL
INTO resultado VALUES(11,90,9,1,6.3,'A',11,2)
INTO resultado VALUES(12,80,8,2,5.6,'A',12,2)
INTO resultado VALUES(13,60,6,4,4.2,'A',13,2)
INTO resultado VALUES(14,50,5,5,3.5,'R',14,2)
INTO resultado VALUES(15,30,3,7,2.1,'R',15,2)
INTO resultado VALUES(16,90,9,1,6.3,'A',16,2)
INTO resultado VALUES(17,90,9,1,6.3,'A',17,2)
INTO resultado VALUES(18,90,9,1,6.3,'A',18,2)
INTO resultado VALUES(19,50,5,5,3.5,'R',19,2)
INTO resultado VALUES(20,30,3,7,2.1,'R',20,2)
SELECT * FROM dual;
```

5-Consultas

```
--Pregunta 1: Conocer el número de evaluaciones por curso
SELECT p.nombre, COUNT(e.programa_idprograma)cantidad_evaluaciones
FROM evaluacion e
RIGHT JOIN programa p
ON e.programa_idprograma = p.idprograma
GROUP BY p.nombre;

--Pregunta 2: Conocer los cursos sin evaluaciones
SELECT p.nombre cursos_sin_evaluaciones
FROM evaluacion e
RIGHT JOIN programa p
ON e.programa_idprograma = p.idprograma
WHERE e.programa_idprograma IS NULL;

--Pregunta 3: Determinar evaluaciones ineficientes
--Una evaluación es deficiente:
--a)Si no tiene preguntas
SELECT e.nombre evaluaciones_sin_preguntas
FROM pregunta p
RIGHT JOIN evaluacion e
ON e.idtest = p.evaluacion_idtest
WHERE p.evaluacion_idtest IS NULL;

--Una evaluación es deficiente:
--a)Si hay preguntas con 2 o menos alternativas
SELECT e.nombre evaluacion_deficiente, p.enunciado pregunta_deficiente, p.idpregunta FROM evaluacion e
INNER JOIN pregunta p
ON e.idtest = p.evaluacion_idtest
INNER JOIN (SELECT a.pregunta_idpregunta, COUNT(a.pregunta_idpregunta) FROM alternativa a GROUP BY
a.pregunta_idpregunta
HAVING COUNT(a.pregunta_idpregunta)<3)a
ON p.idpregunta = a.pregunta_idpregunta;

--Una evaluación es deficiente:
--a)Si todas las alternativas son correctas o todas las alternativas son incorrectas
SELECT e.nombre evaluacion_deficiente, p.enunciado pregunta_deficiente, p.idpregunta, a.estado,
a.idalternativa FROM evaluacion e
INNER JOIN pregunta p
ON e.idtest = p.evaluacion_idtest
INNER JOIN alternativa a
ON p.idpregunta = a.pregunta_idpregunta
WHERE a.pregunta_idpregunta=22 OR a.pregunta_idpregunta=23 OR a.pregunta_idpregunta=24 OR
a.pregunta_idpregunta=25
OR a.pregunta_idpregunta=26 OR a.pregunta_idpregunta=27 OR a.pregunta_idpregunta=28 OR
a.pregunta_idpregunta=29 OR a.pregunta_idpregunta=30;


--Pregunta 4: Determinar cuantos alumnos hay en cada curso
SELECT p.nombre curso, COUNT(e.programa_idprograma)cantidad_alumnos
FROM programa p
LEFT JOIN estudiante e
ON p.idprograma = e.programa_idprograma
GROUP BY p.nombre;

/*Pregunta 5: Obtener el puntaje no normalizado de cada evaluación. El puntaje no
normalizado ha sido definido (requerimiento) como: P = buenas – malas/4. Si un alumno
```

```
no contesta en una pregunta exactamente lo mismo que se ha definido como correcto,
la pregunta cuenta como mala a menos que el alumno haya omitido. */
SELECT e.nombre, e.primerapellido, ((r.cantidadcorrectas - r.cantidadincorrectas) / 4)puntaje_no_normalizado
FROM resultado r
INNER JOIN estudiante e
ON e.idestudiante = r.estudiante_idestudiante;

--Pregunta 6: Obtener el puntaje normalizado, o sea, de 1,0 a 7,0.
SELECT e.nombre, e.primerapellido, r.nota
FROM resultado r
INNER JOIN estudiante e
ON e.idestudiante = r.estudiante_idestudiante;

--Pregunta 7: Nombre de estudiantes de un curso determinado que aprueban una
--evaluación determinada (donde la nota de aprobación mínima es un 4,0).
SELECT e.nombre, e.primerapellido, r.nota aprobados, p.nombre curso
FROM resultado r
INNER JOIN estudiante e
ON e.idestudiante = r.estudiante_idestudiante
INNER JOIN programa p
ON e.programa_idprograma = p.idprograma
WHERE r.nota > 3.9 AND p.idprograma=1;

--Pregunta 8: Nota promedio de los estudiantes de un curso determinado, para una evaluación de terminada.
SELECT  p.nombre curso, e.nombre evaluacion, AVG (nota)promedio_curso
FROM resultado r
INNER JOIN programa p
ON r.evaluacion_idtest = p.idprograma
INNER JOIN evaluacion e
ON r.evaluacion_idtest = e.idtest
GROUP BY evaluacion_idtest, p.nombre, e.nombre;
```