SkillQuest

Guía paso a paso para diseñar una Base de Datos desde cero

Pensamiento Computacional y Programación - Unidad 2: Datos e Información

Idea central

Diseñar una base de datos no es adivinar tablas: es leer un problema, descubrir qué cosas existen, cómo se relacionan y qué información necesita responder la aplicación.

Esta guía usa como ejemplo la prueba SkillQuest: estudiantes, cursos, actividades, experiencia, niveles, insignias, progreso y ranking semanal.

Mapa rápido del proceso

Paso	Acción	Qué debes lograr
1	Leer el contexto	Entender qué debe hacer la aplicación.
2	Encontrar entidades	Buscar las cosas importantes: estudiantes, cursos, actividades, insignias.
3	Definir atributos	Decidir qué datos necesita guardar cada entidad.
4	Crear claves	Elegir una clave primaria para identificar cada fila.
5	Relacionar tablas	Decidir si la relación es 1 a N o N a N.
6	Crear tablas intermedias	Resolver las relaciones muchos a muchos.
7	Evitar redundancia	No repetir textos o datos que deberían estar en otra tabla.
8	Probar el diseño	Ver si la BBDD puede responder preguntas reales de la aplicación.

Regla de oro

Una tabla debe representar una cosa importante del sistema. Si una tabla intenta guardar todo, probablemente está mal diseñada.

1. Leer el problema como diseñador de datos

Antes de escribir tablas, lee el enunciado y subraya dos tipos de palabras:

• **Sustantivos:** cosas que existen en la aplicación, como estudiantes, cursos, actividades e insignias.

• **Verbos o acciones:** cosas que ocurren entre entidades, como inscribirse, completar, ganar, pertenecer y entregar experiencia.

Ejemplo SkillQuest

“Un estudiante puede inscribirse en muchos cursos” no significa que pongamos todos los cursos dentro de estudiantes. Significa que existe una relación entre estudiantes y cursos, y como es muchos a muchos se necesita una tabla intermedia.

2. Encontrar las entidades principales

En SkillQuest aparecen estas entidades principales:

Entidad	Por qué existe	Tabla sugerida
Estudiante	Tiene cuenta, experiencia, nivel y realiza acciones dentro de la app.	estudiantes
Categoría	Agrupa cursos como Programación, Matemática o Ciencias.	categorias
Curso	Contiene actividades y entrega experiencia al completarse.	cursos
Tipo de actividad	Clasifica actividades: alternativa, proyecto, investigación, etc.	tipos_actividad

Entidad	Por qué existe	Tabla sugerida
Actividad	Tarea o desafío dentro de un curso.	actividades
Inscripción / progreso	Guarda que un estudiante está inscrito en un curso y su avance.	inscripciones
Actividad realizada	Guarda puntaje, intentos, fecha y si completó una actividad.	actividades_estudiantes
Insignia	Logro desbloqueable.	insignias
Insignia obtenida	Une estudiantes con insignias ganadas.	estudiantes_insignias
Ranking semanal	Funcionalidad extra para registrar experiencia por semana.	ranking_semanal

3. Definir atributos: qué datos guarda cada tabla

Cada tabla necesita columnas. Primero define la información propia de esa tabla y luego agrega sus claves.

Pregunta útil

Para cada tabla pregúntate: “¿Qué necesito saber de esta cosa para que la aplicación funcione?”

Tabla	Clave primaria	Atributos importantes	Claves foráneas
estudiantes	id_estudiante	nombre, correo, curso_escolar, contrasena, experiencia_total, nivel, fecha_registro	-
categorias	id_categoria	nombre, descripcion	-
cursos	id_curso	nombre, descripcion, dificultad, experiencia_curso	id_categoria
tipos_actividad	id_tipo_actividad	nombre, descripcion	-
actividades	id_actividad	titulo, descripcion, experiencia_actividad	id_curso, id_tipo_actividad
inscripciones	id_inscripcion	estado, porcentaje_avance, fecha_inscripcion, fecha_completado	id_estudiante, id_curso
actividades_estudiantes	id_actividad_estudiante	completada, puntaje_obtenido, intentos, fecha_realizacion	id_estudiante, id_actividad
insignias	id_insignia	nombre, descripcion, condicion	-
estudiantes_insignias	id_estudiante_insignia	fecha_obtenida	id_estudiante, id_insignia
ranking_semanal	id_ranking	semana_inicio, semana_fin, experiencia_ganada, posicion	id_estudiante

Clave primaria y clave foránea

PK significa Primary Key: identifica una fila única. FK significa Foreign Key: conecta una tabla con otra. Ejemplo: cursos.id_categoria apunta hacia categorias.id_categoria.

4. Detectar relaciones entre tablas

Después de encontrar tablas, debes unirlas correctamente. La pregunta más importante es: “¿uno puede tener muchos o muchos pueden tener muchos?”

Relación	Tipo	Cómo se implementa	Ejemplo en SkillQuest
Una categoría tiene muchos cursos	1 a N	La FK va en cursos	cursos.id_categoria
Un curso tiene muchas actividades	1 a N	La FK va en actividades	actividades.id_curso
Un tipo de actividad aparece en muchas actividades	1 a N	La FK va en actividades	actividades.id_tipo_actividad
Un estudiante se inscribe en muchos cursos y un curso tiene muchos estudiantes	N a N	Se crea tabla intermedia	inscripciones
Un estudiante realiza muchas actividades y una actividad puede ser realizada por muchos estudiantes	N a N	Se crea tabla intermedia	actividades_estudiantes
Un estudiante gana muchas insignias y una insignia puede ser ganada por muchos estudiantes	N a N	Se crea tabla intermedia	estudiantes_insignias
Un estudiante puede aparecer muchas semanas en el ranking	1 a N	La FK va en ranking_semanal	ranking_semanal.id_estudiante

5. Resolver relaciones muchos a muchos

Una relación muchos a muchos casi siempre necesita una tabla intermedia. Esa tabla no solo une dos entidades: también puede guardar información propia de esa relación.

Relación N a N	Tabla intermedia	Datos propios de la relación
estudiantes - cursos	inscripciones	estado, porcentaje_avance, fecha_inscripcion, fecha_completado
estudiantes - actividades	actividades_estudiantes	completada, puntaje_obtenido, intentos, fecha_realizacion
estudiantes - insignias	estudiantes_insignias	fecha_obtenida

No lo hagas así

No guardes una columna cursos_inscritos = “SQL, Arte, Inglés” dentro de estudiantes. Eso impide filtrar bien, contar bien y conectar correctamente la información.

6. Elegir tipos de datos adecuados

Los tipos de datos deben representar el tipo de información que quieres guardar.

Tipo	Sirve para	Ejemplo
INT	Números enteros, IDs, niveles, experiencia, intentos	id_estudiante INT, nivel INT
VARCHAR(100)	Textos cortos	nombre VARCHAR(100), correo VARCHAR(100)
VARCHAR(255)	Textos más largos, descripciones	descripcion VARCHAR(255)
DATE	Fechas sin hora	semana_inicio DATE
DATETIME	Fecha y hora exacta	fecha_registro DATETIME
DECIMAL(5,2)	Números con decimales, hasta 999.99	porcentaje_avance DECIMAL(5,2)
BIT	Verdadero o falso	completada BIT

7. Evitar redundancia: normalizar sin complicarse

Una buena base de datos evita repetir información innecesaria. Esto se llama normalizar.

Decisión correcta	Qué evita	Ejemplo
Crear tabla categorias	Repetir “Programación” en muchos cursos	cursos guarda id_categoria, no el texto completo
Crear tabla tipos_actividad	Repetir “Mini proyecto” en cada actividad	actividades guarda id_tipo_actividad
Crear tabla inscripciones	Guardar cursos dentro de estudiantes como texto	Cada inscripción es una fila independiente
Crear tabla estudiantes_insignias	Duplicar columnas insignia1, insignia2, insignia3	Cada insignia ganada es una fila
Guardar progreso en inscripciones	Separar el curso del avance del estudiante	Un curso existe aunque nadie esté inscrito

8. Probar si la base de datos responde preguntas reales

Tu diseño debe servir para responder preguntas útiles. Si no puedes responderlas, probablemente falta una tabla, una columna o una relación.

Pregunta de la aplicación	Tablas que usaría
¿Qué cursos tiene inscritos un estudiante?	estudiantes + inscripciones + cursos
¿Cuál es el porcentaje de avance de un estudiante en cada curso?	inscripciones
¿Qué actividades completó un estudiante y con qué puntaje?	estudiantes + actividades_estudiantes + actividades
¿Qué cursos pertenecen a cada categoría?	categorias + cursos
¿Qué estudiantes terminaron un curso específico?	cursos + inscripciones + estudiantes
¿Qué insignias tiene cada estudiante?	estudiantes + estudiantes_insignias + insignias
¿Quién ganó más experiencia esta semana?	ranking_semanal + estudiantes

9. Integrar el desafío extra: ranking semanal

La imagen de resolución incluye una funcionalidad extra: ranking semanal. Para soportarla se agrega una tabla ranking_semanal.

Columna	Tipo	Uso
id_ranking	INT	Identifica cada registro del ranking.
id_estudiante	INT / FK	Indica qué estudiante aparece en el ranking.
semana_inicio	DATE	Fecha inicial de la semana.
semana_fin	DATE	Fecha final de la semana.
experiencia_ganada	INT	Experiencia ganada durante esa semana.
posicion	INT	Lugar ocupado por el estudiante.

Por qué ranking_semanal no va dentro de estudiantes

Porque un estudiante puede tener muchas semanas de ranking. Si lo guardamos dentro de estudiantes, solo podríamos guardar una semana o tendríamos columnas repetidas como semana1, semana2, semana3.

Método de respuesta recomendado para la prueba

1. Escribe una lista de entidades antes de crear tablas.

2. Dibuja mentalmente las relaciones: 1 a N o N a N.

3. Crea primero las tablas fuertes: estudiantes, categorias, cursos, actividades, insignias.

4. Crea las tablas intermedias: inscripciones, actividades_estudiantes, estudiantes_insignias.

5. Agrega claves primarias en todas las tablas.

6. Agrega claves foráneas donde una tabla dependa de otra.

7. Revisa que no estés repitiendo textos o listas dentro de una sola columna.

8. Prueba tu diseño con al menos cinco preguntas que la aplicación debería responder.

9. Agrega el desafío extra y explica qué tabla o columnas nuevas necesita.

Checklist final antes de entregar

Marca mentalmente: tengo tablas claras, PK en cada tabla, FK donde corresponde, relaciones explicadas, mínimo 3 decisiones contra redundancia, 5 preguntas respondibles y desafío extra integrado.

Errores comunes que debes evitar

• Crear una sola tabla gigante con estudiantes, cursos, actividades, insignias y progreso mezclados.

• Guardar listas dentro de una columna, por ejemplo “curso1, curso2, curso3”.

• Repetir el nombre de la categoría en todos los cursos en vez de crear categorias.

• Olvidar las tablas intermedias en relaciones muchos a muchos.

• Crear claves foráneas sin explicar qué tabla conectan.

• Confundir progreso del curso con datos propios del curso. El curso tiene datos generales; el progreso depende de cada estudiante.

Anexo: diagrama de referencia SkillQuest

Este diagrama muestra una posible solución. No es la única solución posible, pero sí respeta las entidades principales, las relaciones y la funcionalidad extra de ranking semanal.

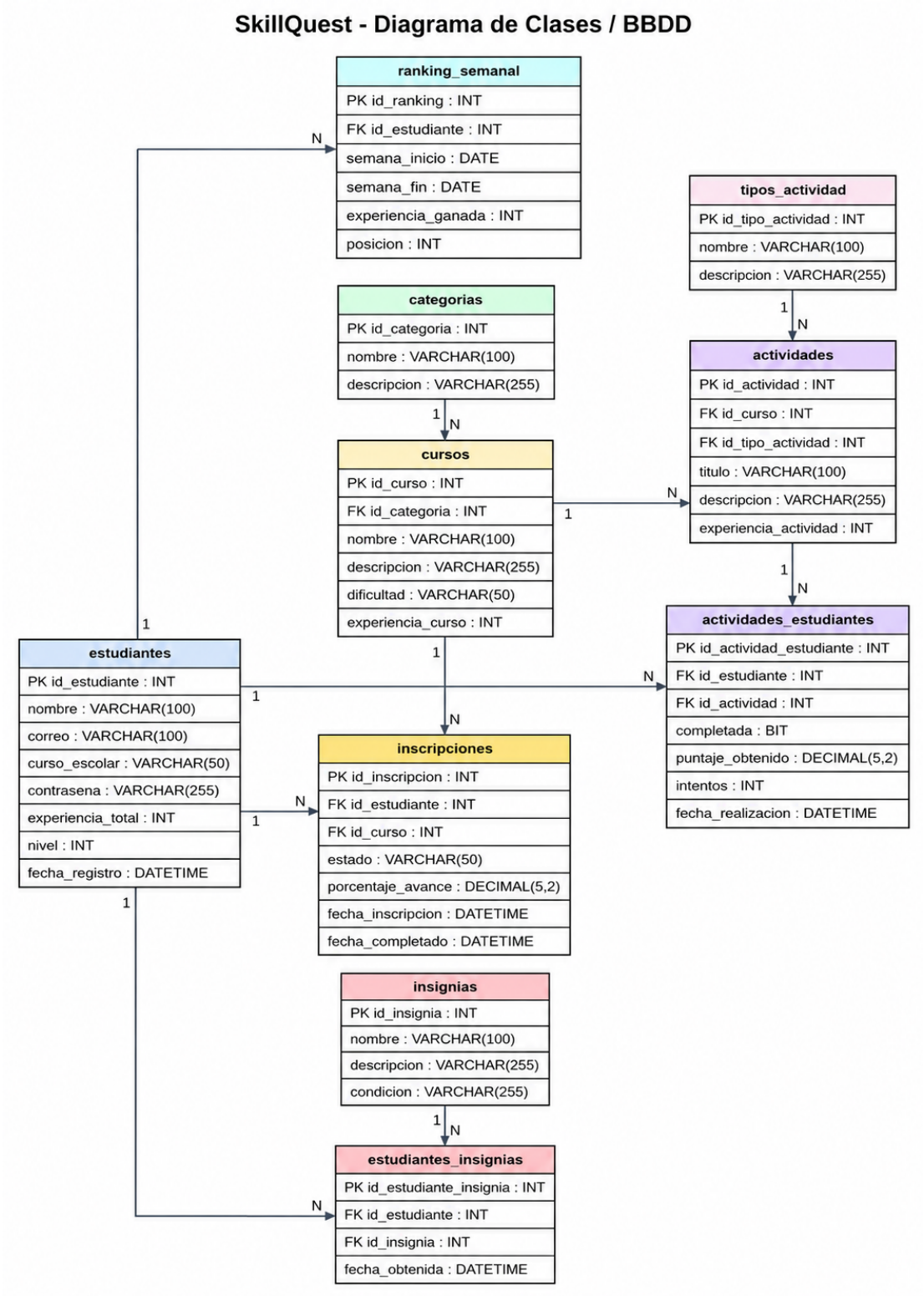

Diagrama de clases / BBDD para SkillQuest.