Banco de Dados Para um Sistema de Auxílio de Atividades Acadêmicas

Camilo Rocha Nascimento, João Alberto Castelo Branco Oliveira, Joel Machado Pires, Matheus Rosa Pithon, Sinezio Morais de Souza Junior, Vinicius Meirelles de Siqueira

Resumo - Gerenciar as componentes curriculares do curso pode parecer uma tarefa fácil se o aluno puder seguir a ordem sugerida no PPC do curso. Entretanto, não é exatamente o que acontece na prática, pois muitos alunos acabam tendo problemas na continuação do curso em algum momento, seja por falta de auxílio durante as matrículas nos semestres anteriores ou diversos outros motivos. Além disso, nem todos sabem como gerenciar a carga horária de disciplinas e/ou de atividades complementares que são exigidas pelo curso. Para isso, foi proposto um banco de dados que possa ser usado em uma aplicação que exibe informações úteis e claras sobre a situação acadêmica de um aluno. Nessa proposta é apresentado modelo entidade-relacionamento, modelo relacional, sua normalização e versão em *PostgreeSQL*, além de uma breve apresentação de aplicação final utilizando esse banco.

Termos—Banco de Dados, Modelo Relacional, Modelo Entidade-Relacionamento, Normalização, Auxílio de Atividades Acadêmicas.

- *Camilo Rocha Nascimento, Universidade Federal do Recôncavo da Bahia, aluno. E-mail: camilorocha11@gmail.com*
- *Joel Machado Pires, Universidade Federal do Recôncavo da Bahia, aluno. E-mail: joelpires70@gmail.com*
- *Matheus Rosa Pithon, Universidade Federal do Recôncavo da Bahia, aluno. E-mail: m.pithon@aluno.ufrb.edu.br*
- *Sinezio Morais de Souza Junior, Universidade Federal do Recôncavo da Bahia, aluno. E-mail: sineziomoras@gmail.com*
- *Vinicius Meirelles de Siqueira, Universidade Federal do Recôncavo da Bahia, aluno. E-mail: vmeirelles.eng@gmail.com*
- *João Alberto Castelo Branco Oliveira, Universidade Federal do Recôncavo da Bahia, professor orientador. E-mail: joaocastelobranco@ufrb.edu.br*

1 Contextualização

A tarefa de gerenciar a grade curricular de um curso de graduação é uma ação de extrema importância para que o discente possa avançar para a conclusão de sua formação. O Censo da educação superior apontou uma taxa de desistência de 56,8% dos alunos em 2018, em uma pesquisa elaborada pelo Ministério da Educação (MEC), sob dados de grupos de alunos universitários que iniciaram os estudos em 2010 [1]. Alguns dos motivos que levam os estudantes a não conseguirem acompanhar a grade proposta pela instituição são: incompatibilidade com o horário do aluno, dificuldades em gerenciar a carga horária de disciplinas e/ou de atividades complementares exigidas pelo curso e reprovações em disciplinas, que dificultam o andamento do curso. Todos esses problemas podem ser referentes à gestão e planejamento da grade do aluno, que muitas vezes não é preparado para lidar com isso [2].

Por outro lado, a computação pervasiva e os sistemas em nuvem têm grande potencial em aplicações voltadas ao ensino [3]–[6]. Entretanto, tais soluções não exploram a problemática apresentada. Nesse sentido, propõe-se um sistema em banco de dados capaz de alimentar uma aplicação web para auxiliar os alunos a gerenciar as disciplinas e carga horária no curso.

Tal sistema irá armazenar o histórico dos alunos que são as matérias já cursadas e calcular quais disciplinas são necessárias a serem cursadas, de forma ordenada, pelos próximos semestres para que o curso flua. Para isso, o banco de dados deve garantir a recuperação, segurança, armazenamento otimizado, integridade e gerenciamento de grandes quantidades de dados lidos ou escritos por diferentes usuários ao mesmo tempo.

Nesse contexto, o sistema deve ser capaz de manipular as informações sobre a graduação do aluno e emitir dados úteis e substanciais que o auxilie a melhor gerenciar seu planejamento acadêmico, aumentando suas chances de finalizar a graduação de maneira bem sucedida.

Portanto, o objetivo é apresentar uma modelagem de banco de dados para o sistema Minha

Grade UFRB.

2 Objetivos

Propor um banco de dados que seja capaz de lidar com informações úteis para auxiliar um aluno a gerenciar o planejamento de sua graduação.

3 Fundamentação Teórica

3.1 Banco de Dados

Aplicações, das mais simples, como sistemas de supermercados, até as mais complexas, como *web services* de redes sociais tratam de dados [7]. Um banco de dados é uma coleção de dados relacionados e os dados são os fatos conhecidos com significados implícitos. É utilizado para representar algum aspecto do mundo real, todo banco é projetado, construído e populado com dados para uma finalidade específica.

Entre um banco de dados e um sistema de gerenciamento de arquivo convencional, o banco se diferencia por possuir natureza de auto-descrição, isolamento entre programas e dados, abstração, suporte de múltiplas visões dos dados, compartilhamento de dados e processamento de transação multiusuário.
Um banco de dados é gerenciado por um Sistema de Gerenciamento de Dados (SGBD) tais como Postgree, MySQL e Oracle. Por exemplo, F. L. ALBINI e P. P. González-Borrero [2010] utilizaram o SGBD Postgree para fazer um sistema web de ensino voltado para física.

3.2 Modelagem de dados usando o modelo Entidade-Relacionamento (ER)

Uma das muitas abordagens para desenvolvimento conceitual de um banco de dados é a modelagem Entidade-Relacionamento (ER), essa modelagem tem a finalidade de colocar o conjunto de requisitos do banco de forma concisa.

Nele, a situação do mundo real é representada através de entidades, relacionamentos e atributos.

As entidades, no diagrama representadas por retângulos, são objetos reais com existência independente. Os atributos, representados por elipses, são as propriedades específicas desse objeto. Para os atributos que os valores são distintos para cada entidade individual, sublinhados no diagrama, o temos como atributo-chave, como forma de identificar unicamente aquele objeto dos demais dentro da mesma entidade. Os relacionamentos, representados por losangos, ditam como essas entidades se vinculam. Eles podem ter restrições de participação e dependências, especificando se a existência de uma entidade depende de sua associação a outra ou não (participação total ou parcial). A restrição total dita a obrigatoriedade de existência de uma das entidades do relacionamento, no diagrama é representado por linhas duplas, como na figura Fig. 1. A restrição parcial restringe a quantidade de uma mesma entidade presente no relacionamento, também chamada de cardinalidade, no diagrama da figura Fig.1., é representado pela numeração ao lado da entidade.

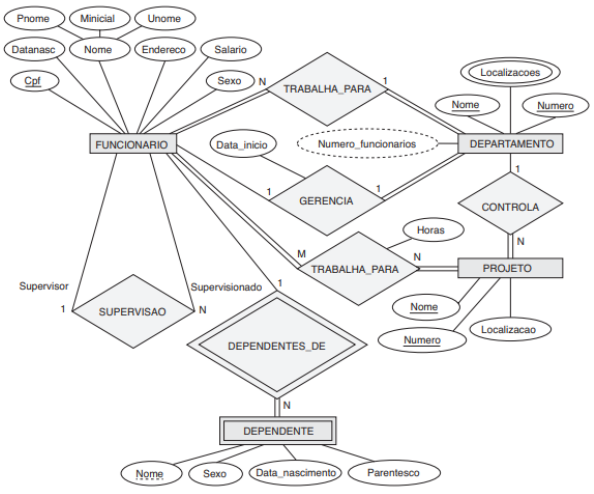

Fig. 1. Diagrama de esquema ER (Entidade Relacionamento) para exemplificar a aplicação em uma empresa genérica. Fonte: [7].

Com esses três itens é possível, então, construir a representação de um mini-mundo.
Um elemento também importante desse modelo é a agregação. Ela é utilizada para indicar um relacionamento entre uma entidade e o que está dentro dela, no diagrama linhas pontilhadas indicam a área da agregação.

3.3 Modelagem de dados usando o modelo Relacional

O modelo relacional representa o banco de dados como uma coleção de relações, também consideradas como tabelas de valores. O modelo relacional deve ter definições lógicas, sobre os domínios dos dados (tipo ou formato). Diferente do modelo ER, nesse modelo não existem entidades, tanto os objetos quanto suas interações são vistos como relações que possuem atributos, exemplificado pela figura Fig. 2.

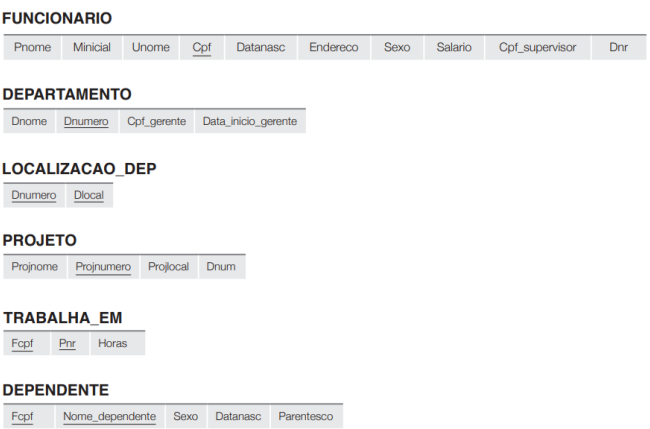

Fig. 2. Diagrama de esquema relacional para exemplificar a aplicação em uma empresa genérica, sem definição de tipos para os atributos. Fonte: [7].

Para os atributos, devem ser feitas as definições lógicas, indicando o seu tipo ou domínio, de maneira a orientar as próximas etapas do desenvolvimento do banco de dados.
Uma relação é definida como um conjunto de tuplas, sem ordem particular ou ordenação, que listam uma

quantidade *n* de valores. Cada tupla na relação pode então ser interpretada como um fato ou uma instância.

3.4 Normalização

O conceito isolado mais importante na teoria de projeto de esquema relacional é o de uma dependência funcional. Uma dependência funcional é uma restrição entre dois conjuntos de atributos do banco de dados. A normalização de dados pode ser considerada um processo de analisar os esquemas de relação dados com base em suas DFs e chaves primárias para conseguir as propriedades desejadas de (1) minimização da redundância e (2) minimização das anomalias de inserção [7].

Edgar F. Codd definiu três tipos de formas normalizadas. A primeira forma normal, em que não há atributos multivalorados no banco de dados. A segunda forma normal em que o modelo está na primeira forma normal e todos os atributos normais dependem exclusivamente da chave primária da tabela. A terceira forma normal em que o modelo está na segunda forma normal e todos os atributos são funcionamento independentes [8].

3.4 Linguagem SQL e SGBD.

À medida que foram surgindo os sistemas de gerenciamento, seus desenvolvedores criaram interfaces próprias para dar acesso aos dados, com o tempo, houve uma necessidade de uma linguagem universal para se interagir com o SGBD e executar suas tarefas. A Linguagem SQL (Structured Query Language) é um padrão de interface criado pelos pesquisadores da IBM, usado para gerenciar bancos de dados relacionais e realizar várias operações nos dados neles, como adicionar e modificar tabelas, atualizar e excluir linhas de dados e etc [9].

Em grande escala, um banco de dados pode conter milhares de informações guardadas, dependendo da estrutura de dados armazenada, uma busca pode durar muito tempo. Para contornar esse problema, o SGBD possibilita buscas otimizadas, através de comandos SQL. A consulta das tabelas é feita por esses comandos que filtram e formatam os dados, usualmente identificados pelas chaves primárias, que são definidas por meio das dependências funcionais.

4. Materiais e Métodos

4.1 Materiais

Com finalidade de realizar o modelo entidade-relação, foi utilizado a aplicação *draw.io*, que é uma ferramenta de código aberto para criação de diagramas online.
Para realização do diagrama relacional, foi utilizado o pacote de ferramentas em linguagem de programação *Python* para criação e visualização de gráficos em código aberto *Graphviz.*

O sistema de gerenciamento de banco de dados utilizado foi *PostgreeSQL,* com todas as requisições sendo realizadas nele.

4.2 Métodos

4.2.1 Conceitualização do Banco

Inicia-se a conceitualização do banco de dados por meio da definição do mini mundo adequado à problemática exposta e identificação dos requerimentos do sistema. Os requerimentos definem quais e como os dados devem ser armazenados para que possam ser requisitados posteriormente. Se tratando de um banco de dados voltado para um sistema de auxílio de atividades acadêmicas, as principais entidades são Usuário, Aluno, Disciplina, Curso, Atividade Complementar e Erro. Com elas, parte-se para a criação do modelo entidade-relacionamento e a definição das restrições e domínio .

4.2.2 Criação do Modelo Entidade-Relacionamento

Para o banco universitário, foram definidas as regras de negócio, e a primeira delas é que o banco será utilizado apenas em uma universidade. Foi definido que a aplicação terá um usuário (Pessoa), e que terá um e-mail, nome e senha. Uma pessoa poderá notificar múltiplos erros. Um erro tem um *id (* identificação), mensagem, título e data. Uma pessoa pode ser múltiplas matrículas (Estudante), onde cada matrícula está associada a *score (pontuação)*, registro, situação de matrícula e semestre inicial. O estudante pode ter atividades complementares, e elas têm registro, título, tipo e valor. O estudante tem que estar registrado em curso que possui nome, código, carga horária obrigatória e optativa. Um curso tem que pertencer a um centro que possui nome e código. Além disso, o curso pode requerer matérias de maneira obrigatória ou optativa em um determinado semestre e com um determinado peso de importância. As matérias têm código, nome e carga horária. Uma matéria requerida em um curso pode requerer outra matéria, como pré requisito. E um aluno pode ter matrícula em uma matéria em um determinado semestre, com um status e *score*.

A representação desse mini mundo é feita por meio do modelo ER e montado com a ferramenta *draw.io* em forma de diagrama.

4.2.3 Criação do Modelo Relacional

O modelo lógico foi construído tendo o modelo entidade-relacionamento como base. Foi levado em consideração as relações das entidades no modelo ER, assim, as relações do modelo relacional foram criadas. Nesta etapa, as entidades tornam-se relações. E as relações que possuem atributos notáveis são colocadas como conjunto de tuplas. Ainda, foi definido o domínio de cada atributo.

4.2.4 Normalização

A transição do modelo ER para o lógico já resulta em um modelo com as dependências funcionais quase que totalmente satisfeitas. Mas ainda restam algumas redundâncias e possíveis anomalias de inserção e atualização. Por isso, foi elevado o modelo à terceira forma normal de Edgar F. Codd.

O modelo idealizado já está na primeira forma normal porque não possui atributos multivalorados e

mais de uma dependência funcional. Ele também já está na segunda forma normal porque não existem dependências parciais. Porém, não se tem o modelo na terceira forma normal porque existe uma dependência transitiva, como mostra a figura Fig. 3.

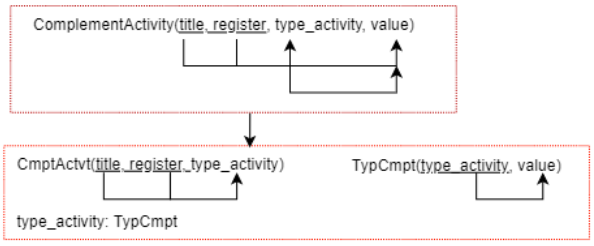

Fig. 3. Dependência transitiva. Fonte: Autoria própria.

Nesta figura Fig. 3, o atributo *value* depende do tipo da atividade complementar (*type_activity*) mas *type_activity* não é chave candidata. Para a normalização, foi separado o atributo que define o tipo, em uma novela tabela *TypCompt* - "Tipo da Atividade Complementar", que possui a dependência funcional *type_activity -> value*. Daí, pode-se ter a relação da atividade complementar (*CmptActvt*) relacionada com o tipo da atividade complementar. Assim, o banco está na terceira forma normal, após essa alteração.

4.2.3 SQL e Requisições

Essa etapa se trata da criação do código em SQL para gerar o banco dentro do SGBD. Foi criado o código baseado no diagrama apresentado no modelo anterior. São definidos aqui as restrição de valores para os atributos, inserindo qual deles podem ou não ser nulos, também foram colocados alguns valores padrão para certos atributos que o banco deve considerar como base sempre que uma nova entidade for inserida.

As requisições definidas como necessárias na conceitualização do banco foram traduzidas de questões para consultas para o banco, de maneira que elas funcionem e retornem o solicitado pelos *PostgreeSQL*.

5 Resultados

5.1 Conceitualização do Banco de Dados

Definiu-se que para atender as necessidades definidas o banco de dados seria responsável por atender as seguintes consultas:

1. *Qual a porcentagem de carga horária executada para o aluno em relação ao total?*
2. *Qual a porcentagem de carga horária obrigatória para o aluno executada em relação ao total?*
3. *Qual a porcentagem de carga horária optativa para o aluno executada em relação ao total?*
4. *Qual a porcentagem de carga horária complementar para o aluno executada em relação ao total?*
5. *Qual a porcentagem de carga horária executada em relação ao total para um semestre em específico?*
6. *Quais matérias já foram pegas pelo aluno?*
7. *Quais as matérias faltam ser pegas pelo aluno?*
8. *Quais as matérias que o aluno não pode pegar?*
9. *Quais matérias o aluno pode pegar?*
10. *Quais as matérias de maior peso do curso?*
11. *Quais as matérias que precisam ser pegas, por ordem de peso?*
12. *Quais as próximas x matérias de maior prioridade?*
13. *Quais as próximas x matérias de maior prioridade que o aluno pode pegar?*
14. *Quantas matérias o aluno está tendo aprovação em média por semestre?*
15. *Qual o índice de aprovação de determinada matéria?*
16. *Qual o índice de aprovação das matérias de determinado curso?*
17. *Qual o índice de aprovação das matérias de determinado centro?*
18. *Com as matérias já executadas pelo aluno neste curso, qual a porcentagem de carga horária obrigatória para o aluno em relação ao total em outro curso?*
19. *Com as matérias já executadas pelo aluno neste curso, qual a porcentagem de carga horária optativa para o aluno executada em relação ao total?*
20. *Com as matérias já executadas pelo aluno neste curso, qual a porcentagem de carga horária complementar para o aluno executada em relação ao total?*
21. *Com as matérias já executadas pelo aluno neste curso, qual a porcentagem de carga horária total para o aluno executada em relação ao total?*
22. *Com as matérias já executadas pelo aluno nesse curso, quais as matérias faltam ser pegas pelo aluno em outro curso*?

5.2 Modelo Entidade-Relacionamento

Seguindo o modelo conceitual apresentado, e as definições para o modelo de negócio, foi obtido o diagrama mostrado pela figura Fig. 4.

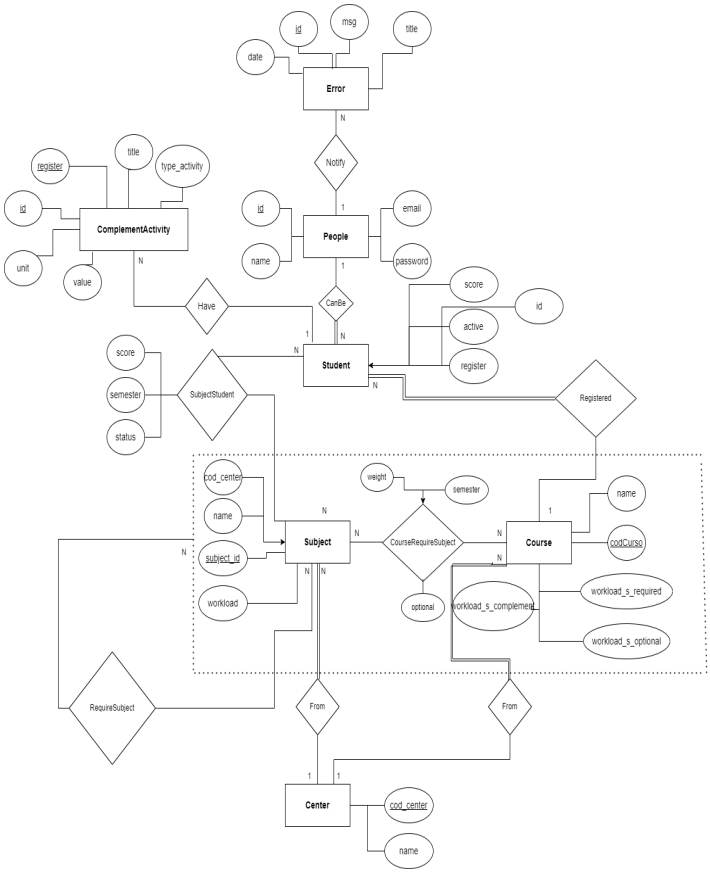

Fig. 4. Modelo ER. Fonte: Autoria própria.

5.3 Modelo Relacional e Normalização

A figura Fig. 5. mostra o modelo relacional junto a normalização realizada.

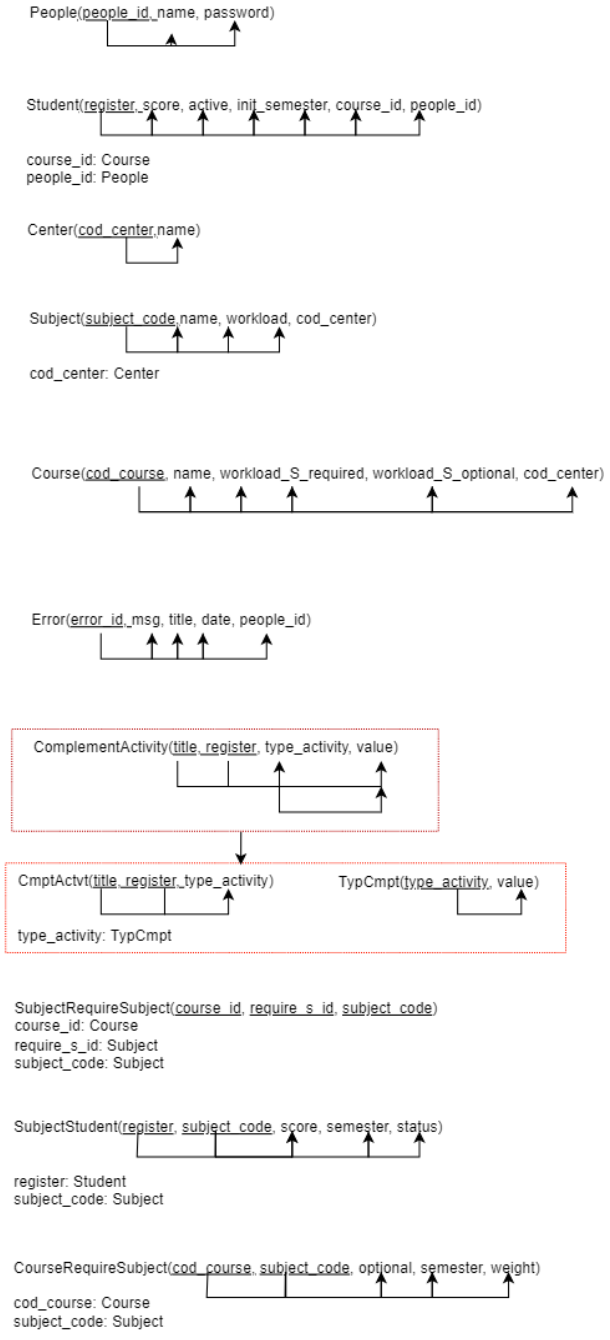

Fig. 5. Modelo relacional e resultados da normalização. Fonte: Autoria própria.

Nessa figura Fig. 5., a origem das setas indicam a chave primária ou composta e o final indicam os atributos dependentes destas chaves. .

5.4 Banco de Dados em SQL e Requisições

Seguindo o modelo relacional apresentado, foi obtido o seguinte código SQL para as tabelas banco:

```
CREATE TABLE "Error" (
    "id" SERIAL NOT NULL,
    "msg" TEXT NOT NULL,
    "title" TEXT NOT NULL,
            "date" TIMESTAMP(3) NOT NULL DEFAULT
CURRENT_TIMESTAMP,
    "email" TEXT NOT NULL,
 CONSTRAINT "Error_pkey" PRIMARY KEY ("id")
);
CREATE TABLE "People" (
    "id" SERIAL NOT NULL,
    "name" VARCHAR(100) NOT NULL,
    "email" VARCHAR(100) NOT NULL,
    "password" TEXT NOT NULL,
 CONSTRAINT "People_pkey" PRIMARY KEY ("email")
);
CREATE TABLE "Student" (
    "id" SERIAL NOT NULL,
    "register" VARCHAR(12) NOT NULL,
    "score" DOUBLE PRECISION NOT NULL DEFAULT 0,
    "active" BOOLEAN NOT NULL DEFAULT true,
    "init_semester" TEXT NOT NULL,
    "email" TEXT NOT NULL,
    "cod_course" TEXT NOT NULL,
CONSTRAINT "Student_pkey" PRIMARY KEY ("register")
);
CREATE TABLE "Center" (
    "id" SERIAL NOT NULL,
    "cod_center" VARCHAR(10) NOT NULL,
    "name" VARCHAR(100) NOT NULL,
CONSTRAINT "Center_pkey" PRIMARY KEY ("cod_center")
);
CREATE TABLE "Course" (
    "id" SERIAL NOT NULL,
    "cod_course" VARCHAR(10) NOT NULL,
    "name" TEXT NOT NULL,
      "workload_S_required" DOUBLE PRECISION NOT NULL
DEFAULT 0,
     "workload_S_optional" DOUBLE PRECISION NOT NULL
DEFAULT 0,
        "workload_S_complement" DOUBLE PRECISION NOT
NULL DEFAULT 0,
    "cod_center" TEXT NOT NULL,
CONSTRAINT "Course_pkey" PRIMARY KEY ("cod_course")
);
CREATE TABLE "Subject" (
    "id" SERIAL NOT NULL,
    "subject_code" VARCHAR(10) NOT NULL,
    "name" VARCHAR(100) NOT NULL,
    "workload" DOUBLE PRECISION NOT NULL DEFAULT 0,
    "cod_center" TEXT NOT NULL,
CONSTRAINT "Subject_pkey" PRIMARY KEY ("subject_code")
);
CREATE TABLE "ComplementActivity" (
    "id" SERIAL NOT NULL,
    "title" TEXT NOT NULL,
    "type_activity" TEXT NOT NULL,
    "register" TEXT NOT NULL,
CONSTRAINT "ComplementActivity_pkey" PRIMARY KEY
("title","register")
);

CREATE TABLE "TypeComplementActivity" (
    "id" SERIAL NOT NULL,
    "type_activity" VARCHAR(100) NOT NULL,
    "value" DOUBLE PRECISION NOT NULL,
CONSTRAINT "TypeComplementActivity_pkey" PRIMARY
```

```
KEY ("type_activity")
);
CREATE TABLE "CourseRequireSubject" (
    "id" SERIAL NOT NULL,
    "optional" BOOLEAN NOT NULL DEFAULT false,
    "semester" TEXT NOT NULL,
    "weight" DOUBLE PRECISION NOT NULL DEFAULT 0,
    "cod_course" TEXT NOT NULL,
    "subject_code" TEXT NOT NULL,
CONSTRAINT "CourseRequireSubject_pkey" PRIMARY KEY ("cod_course","subject_code")
);
CREATE TABLE "SubjectStudent" (
    "id" SERIAL NOT NULL,
    "status" TEXT NOT NULL,
    "score" DOUBLE PRECISION NOT NULL DEFAULT 0,
    "semester" VARCHAR(6) NOT NULL,
    "subject_code" TEXT NOT NULL,
    "register" TEXT NOT NULL,
CONSTRAINT "SubjectStudent_pkey" PRIMARY KEY ("subject_code","register")
);
CREATE TABLE "SubjectRequireSubject" (
    "id" SERIAL NOT NULL,
    "cod_course" TEXT NOT NULL,
    "subject_code" TEXT NOT NULL,
    "require_s_code" TEXT NOT NULL,
CONSTRAINT "SubjectRequireSubject_pkey" PRIMARY KEY ("subject_code","require_s_code","cod_course")
);
AddForeignKey ALTER TABLE "Error" ADD CONSTRAINT "Error_email_fkey" FOREIGN KEY ("email") REFERENCES "People"("email") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "Student" ADD CONSTRAINT "Student_email_fkey" FOREIGN KEY ("email") REFERENCES "People"("email") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "Student" ADD CONSTRAINT "Student_cod_course_fkey" FOREIGN KEY ("cod_course") REFERENCES "Course"("cod_course") ON DELETE RESTRICT ON UPDATE CASCADE; -- AddForeignKey ALTER TABLE "Course" ADD CONSTRAINT "Course_cod_center_fkey" FOREIGN KEY ("cod_center") REFERENCES "Center"("cod_center") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "Subject" ADD CONSTRAINT "Subject_cod_center_fkey" FOREIGN KEY ("cod_center") REFERENCES "Center"("cod_center") ON DELETE RESTRICT ON UPDATE CASCADE; AddForeignKey ALTER TABLE "ComplementActivity" ADD CONSTRAINT "ComplementActivity_register_fkey" FOREIGN KEY ("register") REFERENCES "Student"("register") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "ComplementActivity" ADD CONSTRAINT "ComplementActivity_type_activity_fkey" FOREIGN KEY ("type_activity") REFERENCES "TypeComplementActivity"("type_activity") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "CourseRequireSubject" ADD CONSTRAINT "CourseRequireSubject_cod_course_fkey" FOREIGN KEY ("cod_course") REFERENCES "Course"("cod_course") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "CourseRequireSubject" ADD CONSTRAINT "CourseRequireSubject_subject_code_fkey" FOREIGN KEY ("subject_code") REFERENCES "Subject"("subject_code") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "SubjectStudent" ADD CONSTRAINT "SubjectStudent_subject_code_fkey" FOREIGN KEY ("subject_code") REFERENCES "Subject"("subject_code") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "SubjectStudent" ADD CONSTRAINT "SubjectStudent_register_fkey" FOREIGN KEY ("register") REFERENCES "Student"("register") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "SubjectRequireSubject" ADD CONSTRAINT "SubjectRequireSubject_cod_course_fkey" FOREIGN KEY ("cod_course") REFERENCES "Course"("cod_course") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "SubjectRequireSubject" ADD CONSTRAINT "SubjectRequireSubject_subject_code_fkey" FOREIGN KEY ("subject_code") REFERENCES "Subject"("subject_code") ON DELETE RESTRICT ON UPDATE CASCADE;
AddForeignKey ALTER TABLE "SubjectRequireSubject" ADD CONSTRAINT "SubjectRequireSubject_require_s_code_fkey" FOREIGN KEY ("require_s_code") REFERENCES "CourseRequireSubject"("subject_code") ON DELETE RESTRICT ON UPDATE CASCADE;
```

Por motivos de extensão de código as requisições criadas em código SQL se encontram em apêndice.

6 Conclusão

A metodologia proposta possibilitou a conclusão do desenvolvimento do banco relacional para a aplicação de gerenciamento de carga horária e disciplinas para alunos universitários. Ademais, a solução demonstra potencial contribuição para a sociedade, como diminuição dos índices de reprovações, melhor aproveitamento do curso, redução dos índices de desistência, entre outros. Foram apresentados aspectos importantes para o desenvolvimento de uma boa base de dados e chegar-se à terceira forma normal. Um estudo exploratório com testes com usuários reais pode ser conduzido em trabalhos futuros.

7 Referências

[1] Ministério da Educação, "censo da educação superior 2018 - Divulgação dos resultados." INEP - Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira, 19 de setembro de 2019. Acessado: 3 de agosto de 2022. [Online]. Disponível em: https://download.inep.gov.br/educacao_superior/censo_superior/documentos/2019/apresentacao_censo_superior2018.pdf

[2] M. Bardagi e C. S. Hutz, "Evasão universitária e serviços de apoio ao estudante: uma breve revisão da literatura brasileira", *Psicologia Revista*, vol. 14, nº 2, p. 279–301, 2005.

[3] F. L. ALBINI e P. P. González-Borrero, "Sistema Web de Ensino voltado aos conteúdos da Física", *Sistemas, Cibernética e Informática, Paraná*, vol. 7, nº 2, 2010.

[4] D. F. Da Silva, M. G. Fuini, e J. Schimiguel, "UM

SISTEMA WEB PARA ENSINO DE LÓGICA E ALGORITMOS PARA ALUNOS DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS", em *Proceedings of COPEC World Congress*, 2014, vol. 5.

[5] P. H. Rezende e M. Eloy, "Sistema web para apoio ao ensino de biologia molecular e bioinformática", em *Proceedings of International Conference on Interactive Computer aided Blended Learning*, 2013, p. 365–368.

[6] M. B. de Lima, L. B. Pereira, C. G. Meríno, e M. Struchiner, "Realidade Aumentada no Ensino de Ciências: uma revisão de literatura", 2017.

[7] R. Elmasri, S. B. Navathe, M. G. Pinheiro, e others, "Sistemas de banco de dados", 2005.

[8] E. F. Codd, "Relational completeness of data base sublanguages", em *Database Systems*, 1972, p. 65–98.

[9] R. L. de C. Costa, *SQL Guia Prático-2a edição*. Brasport, 2007.

8 Apêndice

8.1 Sugestão de Aplicação que Utilize o Banco de Dados Apresentado

Com finalidade de demonstração e teste da proposta do banco de dados, foi desenvolvido uma aplicação *web* para comprovar a eficiência do banco de dados. Ela foi realizada utilizando-se a linguagem de programação *Javascript* com aplicação de interfaces *HTML/CSS.* O projeto é código aberto e se encontra disponível em: *https://github.com/GROUP-UFRB/MinhaGradeUFRB*. Essa aplicação realiza as requisições conforme elaboradas no banco de dados e exibe os dados em uma interface gráfica para o usuário final. Para demonstração rápida foi desenvolvida apenas interface básica com pouca população do banco de dados, mas a partir dela já foi possível obter bons resultados no quesito informações úteis a auxílio de estudantes, representada pela figura Fig. 6.

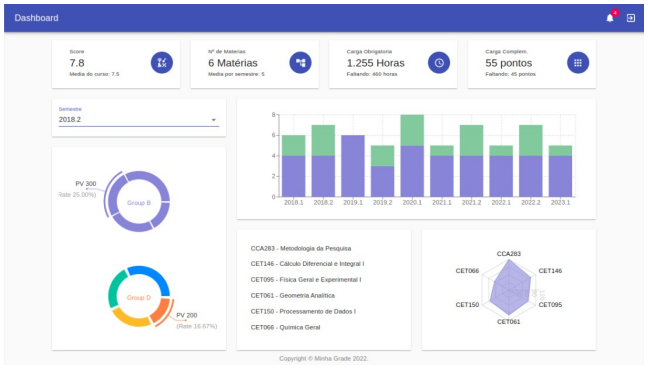

Fig. 6. Apresentação de interface gráfica retornando consultas do banco de dados para o usuário final. Fonte: autoria própria.

8.2 Requisições em SQL

A seguir são apresentadas as requisições criadas para o banco de dados em Linguagem SQL:

```
/*Qual a porcentagem de carga horária executada para o aluno em relação ao total?*/

with total_carga_horaria_curso as (
   SELECT
      sum(workload) as carga_horaria_total_curso
   FROM
      "CourseRequireSubject" crs
      JOIN "Subject" s ON crs.subject_code = s.subject_code
   WHERE
      crs.cod_course = 'BCET'
),
total_carga_horaria_aluno as (
   SELECT
      sum(workload) as carga_horaria_total_aluno
   FROM
      "SubjectStudent" ss
      JOIN "Subject" s ON s.subject_code = ss.subject_code
            JOIN "CourseRequireSubject" crs ON crs.subject_code = s.subject_code
   WHERE
      crs.cod_course = 'BCET'
      and ss.register = '201811509'
      and ss.status = 'aprovado'
)
SELECT
   (
                  cast(carga_horaria_total_aluno as float8) / carga_horaria_total_curso
   ) as porcentagem_carga_horario
FROM
   total_carga_horaria_curso,
   total_carga_horaria_aluno

/* Qual a porcentagem de carga horária obrigatória para o aluno executada em relação ao total? */

with total_carga_horaria_curso as (
  SELECT
    sum(workload) as carga_horaria_total_curso
  FROM
    "CourseRequireSubject" crs
    JOIN "Subject" s ON crs.subject_code = s.subject_code
  WHERE
    crs.cod_course = 'BCET'
    and crs.optional = false
),
total_carga_horaria_aluno as (
  SELECT
    sum(workload) as carga_horaria_total_aluno
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON s.subject_code = ss.subject_code
          JOIN "CourseRequireSubject" crs ON crs.subject_code = s.subject_code
  WHERE
    crs.cod_course = 'BCET'
    and ss.register = '201811509'
    and ss.status = 'aprovado'
    and crs.optional = false
)
SELECT
  (
                  cast(carga_horaria_total_aluno as float8) / carga_horaria_total_curso
  ) as porcentagem_carga_horario_restante
FROM
  total_carga_horaria_curso,
  total_carga_horaria_aluno

/*Qual a porcentagem de carga horária optativa para o aluno executada em relação ao total?*/
```

```
with carga_horaria_opcional_curso as (
  SELECT
    "workload_S_optional" as carga_horaria_opcional_curso
  FROM
    "Course" co
  WHERE
    co.cod_course = 'BCET'
),
carga_horaria_opcional_aluno as (
  SELECT
    sum(workload) as carga_horaria_opcional_aluno
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON s.subject_code = ss.subject_code
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
  WHERE
    crs.cod_course = 'BCET'
    and ss.register = '201811509'
    and ss.status = 'aprovado'
    and crs.optional = true
)
SELECT
  (
                cast(carga_horaria_opcional_aluno as float8) /
carga_horaria_opcional_curso
  ) as porcentagem_carga_horaria_opcional
FROM
  carga_horaria_opcional_curso,
  carga_horaria_opcional_aluno

/*Qual a porcentagem de carga horária complementar para o aluno
executada em relação ao total?*/

with carga_horaria_complementar_curso as (
    SELECT
                                    "workload_S_complement" as
carga_horaria_complementar_curso
    FROM
      "Course" co
    WHERE
      co.cod_course = 'BCET'
),
carga_horaria_complementar_aluno as (
    SELECT
      sum(tca.value) as carga_horaria_complementar_aluno
    FROM
      "ComplementActivity" ca
        JOIN "TypeComplementActivity" tca ON tca.type_activity =
ca.type_activity
    WHERE
      ca.register = '201811509'
)
SELECT
    (
                cast(carga_horaria_complementar_aluno as float8) /
carga_horaria_complementar_curso
    ) as porcentagem_carga_horaria_complementar
FROM
    carga_horaria_complementar_curso,
    carga_horaria_complementar_aluno

/*Qual a porcentagem de carga horária complementar para o aluno
executada em relação ao total?*/

with carga_horaria_complementar_curso as (
    SELECT
                                    "workload_S_complement" as
carga_horaria_complementar_curso
    FROM
      "Course" co
    WHERE
      co.cod_course = 'BCET'
),
carga_horaria_complementar_aluno as (
    SELECT
      sum(tca.value) as carga_horaria_complementar_aluno
    FROM
      "ComplementActivity" ca
        JOIN "TypeComplementActivity" tca ON tca.type_activity =
ca.type_activity
    WHERE
      ca.register = '201811509'
)
SELECT
    (
                cast(carga_horaria_complementar_aluno as float8) /
carga_horaria_complementar_curso
    ) as porcentagem_carga_horaria_complementar
FROM
    carga_horaria_complementar_curso,
    carga_horaria_complementar_aluno

/*Qual a porcentagem de carga horária executada em relação ao total
para um semestre em específico?*/

with total_carga_horaria_curso as (
    SELECT
      sum(workload) as carga_horaria_total_curso
    FROM
      "CourseRequireSubject" crs
      JOIN "Subject" s ON crs.subject_code = s.subject_code
    WHERE
      crs.cod_course = 'BCET'
),
total_carga_horaria_aluno as (
    SELECT
      sum(workload) as carga_horaria_total_aluno
    FROM
      "SubjectStudent" ss
      JOIN "Subject" s ON s.subject_code = ss.subject_code
          JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
    WHERE
      crs.cod_course = 'BCET'
      and ss.register = '201811509'
      and ss.status = 'aprovado'
      and ss.semester = '1'
)
SELECT
    (
                    cast(carga_horaria_total_aluno as float8) /
carga_horaria_total_curso
    ) as porcentagem_carga_horario
FROM
    total_carga_horaria_curso,
    total_carga_horaria_aluno

/*Query: "Quais matérias já foram pegas pelo aluno?" */
select
    *
from
    "SubjectStudent" as ss
where
    ss.register = '201811509'

/*
 query: Quais as matérias faltam ser pegas pelo aluno?
 */
SELECT
          s.name,
```

```
        s.subject_code,
        s.workload
FROM
        "CourseRequireSubject" crs
        JOIN "Subject" s ON crs.subject_code = s.subject_code
EXCEPT
SELECT
        s.name,
        s.subject_code,
        s.workload
FROM
        "SubjectStudent" ss
        JOIN "Subject" s ON ss.subject_code = s.subject_code
WHERE
        ss.status != 'reprovado'
        and ss.register = '201811509'
ORDER BY
        name

/* Quais as matérias que o aluno não pode pegar?*/

with materias_nao_pegas as (
  SELECT
    s.subject_code,
    s.name,
    s.workload,
    crs.weight
  FROM
    "CourseRequireSubject" crs
    JOIN "Subject" s ON s.subject_code = crs.subject_code
  EXCEPT
  SELECT
    s.subject_code,
    s.name,
    s.workload,
    crs.weight
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON s.subject_code = ss.subject_code
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
  WHERE
    ss.register = '201811509'
    and ss.status = 'aprovado'
    and crs.cod_course = 'BCET'
),
id_materias_bloqueadas as (
  SELECT
    srs.require_s_code
  FROM
    "materias_nao_pegas" mnp
        JOIN "SubjectRequireSubject" srs ON srs.subject_code =
mnp.subject_code
),
materias_nao_pode_pegar as (
  /*Materias que não podem ser pegas*/
  SELECT
    distinct *
  FROM
    "id_materias_bloqueadas" r1
    JOIN "Subject" s ON s.subject_code = r1.require_s_code
)
SELECT
  mnpp.subject_code,
  mnpp.name,
  mnpp.workload,
  crs.weight
FROM
  "materias_nao_pode_pegar" mnpp
      JOIN "CourseRequireSubject" crs ON crs.subject_code =
mnpp.subject_code
WHERE
  crs.cod_course = 'BCET'
order by
  name

/* Quais as matérias o aluno pode pegar?*/

with materias_nao_pegas as (
  SELECT
    s.subject_code,
    s.name,
    s.workload,
    crs.weight
  FROM
    "CourseRequireSubject" crs
    JOIN "Subject" s ON s.subject_code = crs.subject_code
  WHERE
    crs.cod_course = 'BCET'
  EXCEPT
  SELECT
    s.subject_code,
    s.name,
    s.workload,
    crs.weight
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON s.subject_code = ss.subject_code
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
  WHERE
    ss.register = '201811509'
    and ss.status = 'aprovado'
    and crs.cod_course = 'BCET'
),
id_materias_bloqueadas as (
  SELECT
    srs.require_s_code
  FROM
    "materias_nao_pegas" mnp
        JOIN "SubjectRequireSubject" srs ON srs.subject_code =
mnp.subject_code
),
materias_nao_pode_pegar as (
  /*Materias que não podem ser pegas*/
  SELECT
    distinct *
  FROM
    "id_materias_bloqueadas" r1
    JOIN "Subject" s ON s.subject_code = r1.require_s_code
)
SELECT
  mnp.subject_code,
  mnp.name,
  mnp.workload,
  mnp.weight
FROM
  "materias_nao_pegas" mnp
EXCEPT
SELECT
  distinct mnpp.subject_code,
  mnpp.name,
  mnpp.workload,
  crs.weight
FROM
  "materias_nao_pode_pegar" mnpp
      JOIN "CourseRequireSubject" crs ON crs.subject_code =
mnpp.subject_code
WHERE
  crs.cod_course = 'BCET'
order by
  name
```

```
/*
 Query: Quais as matérias de maior peso do curso?
 */
/*
 TODO assert provide student_id
 */
SELECT
	sub.subject_code,
	sub.name,
	sub.cod_center,
	crs.weight,
	sub.workload
FROM
	"CourseRequireSubject" crs
	JOIN "Subject" sub ON sub.subject_code = crs.subject_code
WHERE
	crs.cod_course = 'BCET'
ORDER BY
	crs.weight DESC
LIMIT
	5
/*
 Query: Quais as matérias que precisam ser pegas, por ordem de peso?
 */
/*
 TODO assert provide student_id
 */
SELECT
	s.subject_code,
	s.name,
	s.cod_center,
	crs.weight,
	s.workload
FROM
	"CourseRequireSubject" crs
	JOIN "Subject" s ON s.subject_code = crs.subject_code
WHERE
	crs.cod_course = 'BCET'
EXCEPT
SELECT
	s.subject_code,
	s.name,
	s.cod_center,
	crs.weight,
	s.workload
FROM
	"SubjectStudent" ss
	JOIN "Subject" s ON s.subject_code = ss.subject_code
	JOIN "CourseRequireSubject" crs ON s.subject_code =
crs.subject_code
WHERE
	ss.register = '201811509'
	AND ss.status = 'aprovado'
	AND crs.cod_course = 'BCET'
ORDER BY
	weight DESC
/*
 Quais as próximas x matérias de maior prioridade?
 */
SELECT
  s.subject_code,
  s.name,
  s.workload,
  crs.weight
FROM
  "CourseRequireSubject" crs
  JOIN "Subject" s ON s.subject_code = crs.subject_code
EXCEPT
SELECT
  s.subject_code,
  s.name,
  s.workload,
  crs.weight
FROM
  "SubjectStudent" ss
  JOIN "Subject" s ON s.subject_code = ss.subject_code
      JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
WHERE
  ss.register = '201811509'
  and ss.status = 'aprovado'
  and crs.cod_course = 'BCET'
ORDER BY
  weight DESC
LIMIT
  6
  /*x*/
/* Quais as próximas x matérias de maior prioridade que o aluno pode
pegar? */
with materias_nao_pegas as (
  SELECT
    s.subject_code,
    s.name,
    s.workload,
    crs.weight
  FROM
    "CourseRequireSubject" crs
    JOIN "Subject" s ON s.subject_code = crs.subject_code
  WHERE
    crs.cod_course = 'BCET'
  EXCEPT
  SELECT
    s.subject_code,
    s.name,
    s.workload,
    crs.weight
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON s.subject_code = ss.subject_code
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
  WHERE
    ss.register = '201811509'
    and ss.status = 'aprovado'
    and crs.cod_course = 'BCET'
),
id_materias_bloqueadas as (
  SELECT
    srs.require_s_code
  FROM
    "materias_nao_pegas" mnp
      JOIN "SubjectRequireSubject" srs ON srs.subject_code =
mnp.subject_code
),
materias_nao_pode_pegar as (
  /*Materias que não podem ser pegas*/
  SELECT
    distinct *
  FROM
    "id_materias_bloqueadas" r1
    JOIN "Subject" s ON s.subject_code = r1.require_s_code
)
SELECT
  mnp.subject_code,
  mnp.name,
  mnp.workload,
  mnp.weight
FROM
  "materias_nao_pegas" mnp
EXCEPT
SELECT
```

```
    distinct mnpp.subject_code,
    mnpp.name,
    mnpp.workload,
    crs.weight
FROM
    "materias_nao_pode_pegar" mnpp
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
mnpp.subject_code
WHERE
    crs.cod_course = 'BCET'
order by
    weight desc
limit
    6
/*
 Query: Quantas matérias o aluno está tendo aprovação em média por
semestre?
 */
with materias_por_semestre as (
        SELECT
                semester,
                count(ss.subject_code) as materias
        FROM
                "SubjectStudent" ss
                JOIN "Subject" s ON s.subject_code =
ss.subject_code
        WHERE
                ss.register = '201811509'
        GROUP BY
                semester
),
aprovacoes_por_semestre as (
        SELECT
                semester,
                count(ss.subject_code) as aprovacoes
        FROM
                "SubjectStudent" ss
                JOIN "Subject" s ON s.subject_code =
ss.subject_code
        WHERE
                ss.register = '201811509'
                and ss.status = 'aprovado'
        GROUP BY
                semester
)
SELECT
        ma.semester,
        ma.materias,
        ap.aprovacoes,
        (cast(ap.aprovacoes as float) / ma.materias) as indice
FROM
        "materias_por_semestre" ma
        LEFT JOIN "aprovacoes_por_semestre" ap ON ma.semester
= ap.semester/*
 Query: Qual o índice de aprovação de determinada matéria?
 */
with resultados as (
    SELECT
        s.subject_code,
        count(ss.id) as todos_resultados
    FROM
        "CourseRequireSubject" crs
        JOIN "Subject" s ON s.subject_code = crs.subject_code
        JOIN "SubjectStudent" ss ON ss.subject_code = s.subject_code
    WHERE
        crs.cod_course = 'BCET'
        and ss.subject_code = 'CCA310'
        and ss.status != 'cursando'
    GROUP BY
        s.subject_code
),
aprovados as (
    SELECT
        s.subject_code,
        count(ss.id) as todos_aprovados
    FROM
        "CourseRequireSubject" crs
        JOIN "Subject" s ON s.subject_code = crs.subject_code
        JOIN "SubjectStudent" ss ON ss.subject_code = s.subject_code
    WHERE
        crs.cod_course = 'BCET'
        and ss.subject_code = 'CCA310'
        and ss.status = 'aprovado'
    GROUP BY
        s.subject_code
)
SELECT
    re.subject_code,
    (
        cast(todos_aprovados as float) / todos_resultados
    ) as indice_aprovacao
FROM
    "resultados" re,
    "aprovados" ap

/**Qual o índice de aprovação das matérias de determinado curso?*/

with r1 as (
  SELECT
    s.subject_code,
    count(ss.subject_code) as qtd_approved
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON ss.subject_code = s.subject_code
        JOIN "CourseRequireSubject" crs ON s.subject_code =
crs.subject_code
  WHERE
    crs.cod_course = 'BCET'
    and ss.status = 'aprovado'
  GROUP BY
    s.subject_code
),
r2 as (
  SELECT
    s.subject_code,
    count(ss.subject_code) as qtd_registers
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON ss.subject_code = s.subject_code
        JOIN "CourseRequireSubject" crs ON s.subject_code =
crs.subject_code
  WHERE
    crs.cod_course = 'BCET'
    and ss.status != 'cursando'
  GROUP BY
    s.subject_code
)
SELECT
  r2.subject_code,
  Cast(r1.qtd_approved as float) / r2.qtd_registers as indice_aprovacao
FROM
  r2
  LEFT JOIN r1 ON r1.subject_code = r2.subject_code

/**Qual o índice de aprovação das matérias de determinado centro?*/
with r1 as (
  SELECT
    s.subject_code,
    count(ss.subject_code) as qtd_approved
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON ss.subject_code = s.subject_code
```

```
  WHERE
    s.cod_center = 'CETEC'
    and ss.status = 'aprovado'
  GROUP BY
    s.subject_code
),
r2 as (
  SELECT
    s.subject_code,
    count(ss.subject_code) as qtd_registers
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON ss.subject_code = s.subject_code
  WHERE
    s.cod_center = 'CETEC'
    and ss.status != 'cursando'
  GROUP BY
    s.subject_code
)
SELECT
  r2.subject_code,
  Cast(r1.qtd_approved as float) / r2.qtd_registers as indice_aprovacao
FROM
  r2
  LEFT JOIN r1 ON r2.subject_code = r1.subject_code

/* Com as matérias já executadas pelo aluno neste curso,
  qual a porcentagem de carga horária obrigatória para o aluno em
relação ao total em outro curso? */
with total_carga_horaria_curso as (
  SELECT
    sum(workload) as carga_horaria_total_curso
  FROM
    "CourseRequireSubject" crs
    JOIN "Subject" s ON crs.subject_code = s.subject_code
  WHERE
    crs.cod_course = 'BCET'
    /*Curso objetivo*/
    and crs.optional = false
),
total_carga_horaria_aluno as (
  SELECT
    sum(workload) as carga_horaria_total_aluno
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON s.subject_code = ss.subject_code
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
  WHERE
    crs.cod_course = 'BCET'
    /*Curso atual*/
    and ss.register = '201811509'
    and ss.status = 'aprovado'
    and crs.optional = false
)
SELECT
  (
                    cast(carga_horaria_total_aluno as float8) /
carga_horaria_total_curso
  ) as porcentagem_carga_horario_restante
FROM
  total_carga_horaria_curso,
  total_carga_horaria_aluno

/*Com as matérias já executadas pelo aluno neste curso,
 qual a porcentagem de carga horária optativa para o aluno executada
em relação ao total?*/
with carga_horaria_opcional_curso as (
  SELECT
    "workload_S_optional" as carga_horaria_opcional_curso
  FROM
    "Course" co
  WHERE
    co.cod_course = 'BCET'
),
carga_horaria_opcional_aluno as (
  SELECT
    sum(workload) as carga_horaria_opcional_aluno
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON s.subject_code = ss.subject_code
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
  WHERE
    crs.cod_course = 'BCET'
    /*Curso atual*/
    and ss.register = '201811509'
    and ss.status = 'aprovado'
    and crs.optional = true
)
SELECT
  (
                    cast(carga_horaria_opcional_aluno as float8) /
carga_horaria_opcional_curso
  ) as porcentagem_carga_horaria_opcional
FROM
  carga_horaria_opcional_curso,
  carga_horaria_opcional_aluno

/*Qual a porcentagem de carga horária complementar para o aluno
executada em relação ao total?*/
with carga_horaria_complementar_curso as (
    SELECT
                                    "workload_S_complement" as
carga_horaria_complementar_curso
    FROM
      "Course" co
    WHERE
      co.cod_course = 'BCET'
      /*curso objetivo*/
),
carga_horaria_complementar_aluno as (
    SELECT
      sum(tac.value) as carga_horaria_complementar_aluno
    FROM
      "ComplementActivity" ca
        JOIN "TypeComplementActivity" tac ON tac.type_activity =
ca.type_activity
    WHERE
      ca.register = '201811509'
)
SELECT
    (
                    cast(carga_horaria_complementar_aluno as float8) /
carga_horaria_complementar_curso
    ) as porcentagem_carga_horaria_complementar
FROM
    carga_horaria_complementar_curso,
    carga_horaria_complementar_aluno

/*Com as matérias já executadas pelo aluno neste curso,
 qual a porcentagem de carga horária total para o aluno executada em
relação ao total?*/

with total_carga_horaria_curso as (
    SELECT
      sum(workload) as carga_horaria_total_curso
    FROM
      "CourseRequireSubject" crs
      JOIN "Subject" s ON crs.subject_code = s.subject_code
    WHERE
      crs.cod_course = 'BCET'
```

```
    /*codigo do outro curso*/
),
total_carga_horaria_aluno as (
  SELECT
    sum(workload) as carga_horaria_total_aluno
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON s.subject_code = ss.subject_code
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
  WHERE
    crs.cod_course = 'BCET'
    /*Curso atual*/
    and ss.register = '201811509'
    and ss.status = 'aprovado'
)
SELECT
  (
                    cast(carga_horaria_total_aluno as float8) /
carga_horaria_total_curso
  ) as porcentagem_carga_horario_restante
FROM
  total_carga_horaria_curso,
  total_carga_horaria_aluno

/*Com as matérias já executadas pelo aluno neste curso,
 qual a porcentagem de carga horária total para o aluno executada em
relação ao total?*/
with total_carga_horaria_curso as (
  SELECT
    sum(workload) as carga_horaria_total_curso
  FROM
    "CourseRequireSubject" crs
    JOIN "Subject" s ON crs.subject_code = s.subject_code
  WHERE
    crs.cod_course = 'BCET'
),
total_carga_horaria_aluno as (
  SELECT
    sum(workload) as carga_horaria_total_aluno
  FROM
    "SubjectStudent" ss
    JOIN "Subject" s ON s.subject_code = ss.subject_code
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
  WHERE
    ss.register = '201811509'
    and ss.status = 'aprovado'
)
SELECT
  (
                    cast(carga_horaria_total_aluno as float8) /
carga_horaria_total_curso
  ) as porcentagem_carga_horario_restante
FROM
  total_carga_horaria_curso,
  total_carga_horaria_aluno

/*Quantos alunos estão matriculados em determinada matéria no
semestre atual?*/
SELECT
  *
FROM
  "SubjectStudent" ss
WHERE
  ss.status = 'cursando'
  and ss.subject_code = 'CET095'
/*Quantos alunos estão com matrícula ativa em determinado curso?*/

SELECT
        *
FROM
        "Student" s
WHERE
        s.active = true

/*Quais são as disciplinas que mais possuem reprovações?*/

SELECT
        subject_code, COUNT(subject_code) as reprovacoes
FROM
        "SubjectStudent" ss
WHERE
        ss.status = 'reprovado'
GROUP BY subject_code
ORDER BY reprovacoes DESC
/*Qual a porcentagem de aprovação em uma determinada disciplina?*/
with matriculas as (
  SELECT
    ss.subject_code,
    count(ss.subject_code) as matriculas
  FROM
    "SubjectStudent" ss
  WHERE
    ss.subject_code = 'CET171'
  GROUP BY
    ss.subject_code
),
aprovados as (
  SELECT
    ss.subject_code,
    count(ss.subject_code) as aprovacoes
  FROM
    "SubjectStudent" ss
  WHERE
    ss.status = 'aprovado'
    and ss.subject_code = 'CET171'
  GROUP BY
    ss.subject_code
)
SELECT
  ap.subject_code,
  (cast(aprovacoes as float) / matriculas) as indice_aprovacao
FROM
  "aprovados" ap,
  "matriculas"
/* Para cada matéria, qual a quantidade de alunos que não pegou mas
pode pegar? */
with materias_nao_pegas as (
  SELECT
    s.subject_code,
    s.name,
    s.workload,
    crs.weight,
    st.register
  FROM
    "CourseRequireSubject" crs
    JOIN "Subject" s ON s.subject_code = crs.subject_code
    JOIN "Course" c ON c.cod_course = crs.cod_course
    JOIN "Student" st ON st.cod_course = c.cod_course
  WHERE
    crs.cod_course = 'BCET'
    and st.cod_course = 'BCET'
  EXCEPT
  SELECT
    s.subject_code,
    s.name,
    s.workload,
    crs.weight,
    ss.register
  FROM
    "SubjectStudent" ss
```

```
    JOIN "Subject" s ON s.subject_code = ss.subject_code
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
s.subject_code
  WHERE
    ss.status = 'aprovado'
    and crs.cod_course = 'BCET'
),
id_materias_bloqueadas as (
  SELECT
    srs.require_s_code,
    mnp.register
  FROM
    "materias_nao_pegas" mnp
        JOIN "SubjectRequireSubject" srs ON srs.subject_code =
mnp.subject_code
),
materias_nao_pode_pegar as (
  /*Materias que não podem ser pegas*/
  SELECT
    distinct *
  FROM
    "id_materias_bloqueadas" r1
    JOIN "Subject" s ON s.subject_code = r1.require_s_code
),
retorno as (
  SELECT
    mnp.subject_code,
    mnp.name,
    mnp.workload,
    mnp.weight,
    mnp.register
  FROM
    "materias_nao_pegas" mnp
  EXCEPT
  SELECT
    distinct mnpp.subject_code,
    mnpp.name,
    mnpp.workload,
    crs.weight,
    mnpp.register
  FROM
    "materias_nao_pode_pegar" mnpp
        JOIN "CourseRequireSubject" crs ON crs.subject_code =
mnpp.subject_code
  WHERE
    cod_course = 'BCET'
  order by
    register,
    weight desc
)
SELECT
  subject_code,
  count(register) as quantidade
FROM
  retorno
GROUP BY
  subject_code
```